

# Stochastic Methods for Emulation, Calibration and Reliability Analysis of Engineering Models

Thesis submitted in accordance with the requirements of  
the University of Liverpool for the degree of  
Doctor in Philosophy

by

**Alfredo Garbuno Iñigo**



May 2018



---

## Declaration of Authorship

---

I, Alfredo Garbuno Iñigo, declare that this thesis titled *Stochastic Methods for Emulation, Calibration and Reliability Analysis of Engineering Models*, and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always give. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where this thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I contributed myself.

Signed: \_\_\_\_\_

Date: \_\_\_\_\_





---

## Abstract

---

This dissertation examines the use of non-parametric Bayesian methods and advanced Monte Carlo algorithms for the emulation and reliability analysis of complex engineering computations. Firstly, the problem lies in the reduction of the computational cost of such models and the generation of posterior samples for the Gaussian Process' (GP) hyperparameters. In a GP, as the flexibility of the mechanism to induce correlations among training points increases, the number of hyperparameters increases as well. This leads to multimodal posterior distributions. Typical variants of MCMC samplers are not designed to overcome multimodality. Maximum posterior estimates of hyperparameters, on the other hand, do not guarantee a global optimiser. This presents a challenge when emulating expensive simulators in light of small data. Thus, new MCMC algorithms are presented which allow the use of full Bayesian emulators by sampling from their respective multimodal posteriors. Secondly, in order for these complex models to be reliable, they need to be robustly calibrated to experimental data. History matching solves the calibration problem by discarding regions of input parameters space. This allows one to determine which configurations are likely to replicate the observed data. In particular, the GP surrogate model's probabilistic statements are exploited, and the data assimilation process is improved. Thirdly, as sampling-based methods are increasingly being used in engineering, variants of sampling algorithms in other engineering tasks are studied, that is reliability-based methods. Several new algorithms to solve these three fundamental problems are proposed, developed and tested in both illustrative examples and industrial-scale models.



*To both my stars Adriana and Amelia.*



---

## Acknowledgements

---

First and foremost, I would like to thank my supervisor, Dr Alejandro Diaz De la O, for his guidance and patience during my PhD studies. I am grateful for his academic guidance and for showing me a way in the world of scientific research.

I would like to thank Prof. Ian Vernon and Dr Peter Green for taking the time to read this dissertation. Their insight and comments helped me improve its readability.

I also want to gratefully acknowledge the support of Mexico's National Council for Science and Technology (CONACYT) for funding my studies and allowing me to accomplish this life-long goal of mine.

My studies at the University of Liverpool would not have been the same without the incredible colleagues I met at the Institute for Risk and Uncertainty. I have the pleasure to call them my friends. In particular, special thanks to Peter Hristov, Dominic Calleja, Paul Byrnes, Mathew Edison, Ryan Jackson and Rachel Gong for their companionship in this journey.

Thankfully not everything is work and responsibilities, so I would like to extend my thanks to my Liverpool brothers and sisters: Lulu Guzman Castillo, Simone Bianchi, Eduardo Blanco Davis, David Rico Sierra, Mariana Hernandez, Plami Hristova, and David Vega Herrera for all their support and love towards me and my family.

Incidentally, I would like to thank my family and friends for all their love and

---

support. In particular, to my mother, Nora Iñigo Contreras, my uncles: Victor, Jorge and German, and of course *mis abuelitos*, Yolanda Contreras Ayala and Jorge Alberto Iñigo Camacho<sup>†</sup>. I also want to express my gratitude for the love and support of Adriana Barud Bejos.

Finally, I would like to express my deepest love and gratitude towards Adriana Rodriguez Barud and Amelia Garbuno Rodriguez. Their love inspires me to work hard, laugh, enjoy life, and to be the best researcher/husband/father I can.

---

## Contents

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Aims and objectives . . . . .	5
1.3	Surrogate Modelling . . . . .	6
1.3.1	Regression and interpolation-based models . . . . .	7
1.3.2	High-dimensional model representation . . . . .	12
1.4	History Matching . . . . .	15
1.5	Reliability Analysis . . . . .	18
1.6	Areas of Opportunity . . . . .	19
1.7	Layout of the Dissertation . . . . .	20
<b>2</b>	<b>Preliminaries</b>	<b>23</b>
2.1	Bayesian Inference . . . . .	24
2.1.1	Prior distributions . . . . .	26
2.1.2	Prediction . . . . .	27
2.1.3	Reporting inferences and Markov Chain Monte Carlo . . . . .	28
2.2	Gaussian Processes . . . . .	30
2.2.1	Priors on random linear functions . . . . .	31
2.2.2	The Gaussian process for regression . . . . .	34
2.3	Discussion . . . . .	40

<b>3</b>	<b>Gaussian Process Hyperparameters: PAIMS</b>	<b>43</b>
3.1	The Gaussian Process as an Emulator . . . . .	47
3.1.1	Estimating the Hyperparameters . . . . .	49
3.1.2	Prior Distributions . . . . .	51
3.1.3	Marginalising the Nuisance Hyperparameters . . . . .	52
3.2	AIMS Framework . . . . .	55
3.2.1	Annealing at Level $k$ . . . . .	56
3.2.2	Adaptive Proposal Distribution and Temperature Scheduling	58
3.2.3	Stopping Condition . . . . .	60
3.2.4	Parallel Implementation and Guarding Against Rejection .	61
3.3	Implementation Aspects . . . . .	64
3.4	Numerical Experiments using PAIMS . . . . .	66
3.4.1	Branin Function . . . . .	67
3.4.2	2D Model . . . . .	70
3.4.3	Nilson-Kuusk Model . . . . .	71
3.5	Discussion on PAIMS Results . . . . .	76
3.6	Conclusion . . . . .	77
3.A	Appendix . . . . .	78
<b>4</b>	<b>Gaussian Process Hyperparameters: TA<sup>2</sup>S<sup>2</sup></b>	<b>81</b>
4.1	Slice Sampling . . . . .	82
4.1.1	Adaptive Slice Sampling . . . . .	84
4.2	Transitional Annealed Adaptive Slice Sampling . . . . .	85
4.2.1	Annealing at Level $k$ . . . . .	86
4.2.2	Overview of the Full Sampler - TA <sup>2</sup> S <sup>2</sup> . . . . .	89
4.3	Numerical Experiments using TA <sup>2</sup> S <sup>2</sup> . . . . .	92
4.3.1	Franke's function . . . . .	94
4.3.2	Nilson-Kuusk model . . . . .	97
4.3.3	Wing weight model . . . . .	98
4.4	Discussion on TA <sup>2</sup> S <sup>2</sup> Results . . . . .	100
4.5	Industrial application . . . . .	103
4.6	Conclusion . . . . .	106



<b>5</b>	<b>Probabilistic Extensions to History Matching</b>	<b>107</b>
5.1	History matching . . . . .	110
5.1.1	Probabilistic History Matching . . . . .	112
5.2	NROY space identification . . . . .	113
5.3	Sequential non-implausible design . . . . .	115
5.4	Numerical experiments . . . . .	120
5.4.1	Franke's function . . . . .	120
5.4.2	The IC fault model . . . . .	122
5.4.3	Random functions experimental setting . . . . .	124
5.5	Discussion . . . . .	129
5.6	Industrial application . . . . .	130
5.7	Conclusion . . . . .	132
5.A	Expected risk . . . . .	134
5.B	Entropic profile . . . . .	135
5.C	Random functions from a Gaussian process prior . . . . .	135
<b>6</b>	<b>Reliability Analysis and Hamiltonian Dynamics</b>	<b>137</b>
6.1	Introduction . . . . .	138
6.2	Reliability Analysis and Subset Simulation . . . . .	140
6.2.1	Reliability Analysis . . . . .	140
6.2.2	Subset Simulation . . . . .	141
6.2.3	MCMC Proposals for Subset Simulation . . . . .	142
6.3	Hamiltonian Monte Carlo . . . . .	144
6.3.1	HMC for Subset Simulation . . . . .	146
6.3.2	Smooth failure domains . . . . .	149
6.4	Numerical experiments . . . . .	151
6.4.1	Linear performance function . . . . .	152
6.4.2	Multi-dimensional ring . . . . .	153
6.4.3	Inhomogeneous bar . . . . .	156
6.5	Discussion . . . . .	158
6.6	Conclusion . . . . .	159
<b>7</b>	<b>Reliability-based Calibration</b>	<b>161</b>
7.1	Introduction . . . . .	161

7.2	BUS formulation . . . . .	166
7.2.1	Rejection Principle . . . . .	166
7.2.2	Equivalent reliability problem . . . . .	167
7.3	Likelihood multiplier . . . . .	168
7.4	Alternative BUS formulation . . . . .	171
7.5	Bayesian model class selection . . . . .	174
7.6	Characteristic trends . . . . .	175
7.7	Automatic Stopping Strategy . . . . .	176
7.7.1	Stopping criterion . . . . .	177
7.7.2	Posterior statistical estimation . . . . .	180
7.7.3	Statistical error assessment . . . . .	180
7.7.4	Comparison with original BUS formulation . . . . .	181
7.8	Numerical Experiments . . . . .	183
7.8.1	Example 1. Two-DOF shear frame: locally identifiable case . . . . .	183
7.8.2	Example 2. Two-DOF shear frame: unidentifiable case . . . . .	186
7.8.3	Example 3. Model Class Selection . . . . .	187
7.9	Discussion . . . . .	189
7.10	Conclusion . . . . .	189
<b>8</b>	<b>Summary and Conclusions</b>	<b>191</b>
8.1	Summary of Completed Work . . . . .	192
8.2	Summary of Contributions . . . . .	194
8.3	Outlook . . . . .	196
8.4	Published journal papers . . . . .	196
8.5	Journal Papers Under Review . . . . .	197

---

## List of Figures

---

2.1	Bayesian inference examples. . . . .	26
2.2	Prior assumptions on a linear regression model. . . . .	32
2.3	Posterior model for a simple linear regression. . . . .	33
2.4	Covariance kernels examples. . . . .	37
2.5	Gaussian process example. . . . .	39
3.1	Gaussian process sensitivity to lengthscale hyperparameter. . . . .	48
3.2	Marginal contour levels for scale hyperparameters of a Gaussian process. . . . .	53
3.3	Gaussian process' likelihood and numerical instability. . . . .	65
3.4	Contour levels of Branin function . . . . .	68
3.5	Results of PAIMS in a 2D model. . . . .	69
3.6	Contour levels of the 2D Model . . . . .	71
3.7	Comparing posterior contour levels for 2D example with different priors. . . . .	72
3.8	Results for 2D test function. . . . .	73
3.9	Posterior samples for Nilson-Kuusk model. . . . .	75
3.10	Residuals plot diagnostic for the Nilson-Kuusk simulator. . . . .	76
4.1	Contour levels of Franke's function . . . . .	95
4.2	Posterior samples with TA <sup>2</sup> S <sup>2</sup> algorithm. . . . .	96
4.3	Results using TA <sup>2</sup> S <sup>2</sup> and other sampling alternatives. . . . .	97

4.4	Results for TA <sup>2</sup> S <sup>2</sup> algorithm in Nilson-Kuusk simulator. . . . .	99
4.5	Results for a 10D wing-weight model. . . . .	101
4.6	Emulated responses for gas cloud formation in offshore platforms .	104
4.7	Validation diagnostics for the GP emulator in the gas formation simulator . . . . .	105
5.1	Comparison of pointwise and probabilistic implausibility function	114
5.2	Torus implausibility exercise for multimodal sampling . . . . .	116
5.3	History matching waves for Franke’s function . . . . .	122
5.4	History matching summaries for Franke’s function . . . . .	123
5.5	History matching summaries for the IC fault model . . . . .	125
5.6	Maximum predicted summaries for testbed of functions . . . . .	127
5.7	CRPS summaries for tested of functions . . . . .	128
5.8	NROY identification for the Airbus aircraft design optimisation problem . . . . .	132
6.1	Typical set of multivariate Gaussian and comparison of Metropolis Hastings variants . . . . .	148
6.2	Sigmoid approximation to an indicator function . . . . .	150
6.3	Summary of predicted failure probabilities for linear response . . .	153
6.4	Summary of coefficient of variation for linear response . . . . .	154
6.5	Hamiltonian-based Subset simulation samples for the ring problem	155
6.6	Summary of predicted failure probabilities for the ring problem . .	155
6.7	Summary of coefficient of variation for the ring problem . . . . .	156
6.8	Graphical representation of the inhomogeneous bar. . . . .	156
6.9	Summary of results for the inhomogeneous bar problem . . . . .	157
7.1	Likelihood multiplier and truncation in the BUS context. . . . .	170
7.2	Characteristic trends of $\ln P(Y > b)$ and $V(b)$ . . . . .	176
7.3	Illustration of increasing failure levels and likelihood. . . . .	179
7.4	Sampling results for the automatic BUS algorithm. . . . .	184
7.5	Marginal histograms in the identifiable stiffness problem. . . . .	185
7.6	Empirical log-cumulative probability function and evidence esti- mation. . . . .	186
7.7	Scatter plots of main stiffness parameters for the nonidentifiable example. . . . .	187

7.8	Scatter plots of additional stiffness parameters for the nonidentifiable example. . . . .	188
7.9	Ratio of the identifiable posterior evidence against the nonidentifiable case. . . . .	188



# CHAPTER 1

---

## Introduction

---

### 1.1 Motivation

The use of complex mathematical models has become ubiquitous in modelling manufactured and physical processes in Engineering. Computational representations of such models allow one to bypass the limitations associated with expensive or infeasible experimental designs. In particular, the analysis of computer code output allows the analyst to improve the understanding of the processes under study, identify theoretical shortcomings, or improve a decision-making scenario. In the literature, the computational implementation of a mathematical model is known as a *simulator* [171]. A mathematical model often requires parameters that govern the dynamics and laws of the process. In contrast, simulators use some parameters to represent physical properties of the system; some to compensate for approximations in the simulator (tuning parameters); and some account for external actions influencing the process under study (control parameters) [99]. Thus, the output from such a simulator can be thought as a function  $\eta : \mathcal{X} \rightarrow \mathcal{Y}$ , where  $\mathcal{X} \subseteq \mathbb{R}^d$  and  $\mathcal{Y} \subseteq \mathbb{R}^m$ . In this context,  $\mathcal{X}$  denotes the set of all possible input configurations for the simulator, and  $\mathcal{Y}$  denotes the set of possible output values.

The framework of uncertainty quantification (UQ) allows one to compare the

output of a simulator with the corresponding observed real process. Moreover, through UQ it is possible to identify and quantify uncertainties generated from different sources in the modelling pipeline. The main sources of uncertainty arise from models, numerical algorithms, experiments, and observations of real world processes [199]. In particular, *model uncertainty* arises from the discrepancy of the mathematical representation of the process and the unknown input configuration needed to be used for the simulator. The model itself is built upon assumptions and the exact underlying physics of the process are not known. On the other hand, input values for the simulator cannot truly be observed. At the same time, a chosen simulator input  $\mathbf{x} \in \mathcal{X}$ , also contains some simulator parameters that cannot be observed directly. *Experimental uncertainty* arises from partially observed data and the limited accuracy of measuring devices. This kind of uncertainty is due to the experimental design being a surrogate of a potentially physically infeasible setting, *e.g.* placing sensors in the fuselage of an aircraft. Lastly, *numerical uncertainty* arises from the representation of the model in a computational framework. For example, models based on dynamic systems require approximation methods to be solvable. Numerical uncertainty is also known as *code uncertainty*. These sources of uncertainty can be further classified according to the degree on which they are inherent to the application or reflect pure lack of knowledge. Smith [199] provides the following definitions which are well known in the engineering literature.

**Definition 1** (Aleatoric uncertainty). Also known as statistical, stochastic, or irreducible uncertainty. This is uncertainty inherent to a problem or experiment that in principle cannot be reduced by additional physical or experimental knowledge.

**Definition 2** (Epistemic uncertainty). This is uncertainty due to simplifying model assumptions, missing physics, or basic lack of knowledge.

Aleatoric uncertainty can be defined in a probabilistic framework, whereas epistemic uncertainty is less prone to such a treatment. Hence, one goal of uncertainty quantification is to restate epistemic uncertainties as aleatoric uncertainties where a probabilistic handling is applicable [199, Section 1.1.4]. This classification of uncertainty is just a modelling tool, as with perfect knowledge both types of uncertainty are not needed. In applications, epistemic uncertainty is considered as a type of uncertainty that can be reduced. On the other hand, aleatoric uncertainty represents a type of uncertainty that is infeasible to reduce further. The



probabilistic framework has been the most widespread choice to treat uncertainty. Although it is important to note that imprecise probabilities, Dempster-Shafer Belief functions, possibility measures, and ranking functions are other modelling alternatives [see 109, 219, for an overview]. Bayesian inference is used in this dissertation as it provides the only coherent framework in which uncertainty is treated probabilistically. This allows one to naturally incorporate both types of uncertainty, making this distinction artificial from a Bayesian perspective. This is a consequence of modelling probabilities as a reflection of the decision-maker's state of knowledge about an uncertain event. By using Bayesian theory, one is able to incorporate epistemic uncertainty into the modelling procedure and update probabilities in light of observed data [96].

Framed within the probabilistic framework for uncertainty quantification, simulators are used to propagate input uncertainty to study the implied randomness in the computer code output. However, this process relies on a well calibrated computer model and intensive use of computational resources for experimentation. The *calibration* problem, arises from the need for a model to be able to represent truthfully the process for which it was designed for. The physics, although correct, incorporate parameters that govern the model which are not known with certainty. This motivates the need to find an *appropriate* input configuration, say  $\mathbf{x}^*$ , that will enable a good understanding of the process through the simulator output  $\eta(\mathbf{x}^*)$ . Once  $\mathbf{x}^*$  has been identified, the exploration of its probabilistic neighbourhood allow the analyst to answer questions of interest. In this setting, uncertainty in the output of the model is usually a consequence of uncertainty in the input configuration. In applications such as climate models, nuclear reactor models, or biological models, it is paramount to make predictions with quantified uncertainties. For this purpose, input uncertainty is propagated to the output through extensive simulator runs. However, even though computer power has increased, the computational complexity of the models has increased as well, leading to expensive computer codes infeasible for repeated evaluations. Thus, a full characterisation of the uncertainty in the simulator's output is prohibitive and the need for *surrogate* models arises. A surrogate model aims to retain similar statistical properties from the simulator, at less computational expense.

The Gaussian stochastic process (GP) has been successfully used as a surrogate for computationally expensive computer codes. As such, a Gaussian process not only provides both a surrogate for the simulator, but a full probabilistic

characterisation of the computer code output as well. The reason is that it also provides an internal measure of uncertainty about its predicted value. Therefore, the Gaussian process can be used as an *emulator* for large scale computer code analysis [193]. The original use of the Gaussian process in the context of Uncertainty Quantification dates back to modelling ore reserves in mining [137]. The technique is known as kriging, and its formulation was motivated by finding the best linear unbiased predictor subject to a known covariance structure. Kriging has been widely applied in geostatistics as a predictive model with limited data [51, 50, 204]. According to Owen et al. [175], the first application of Gaussian processes in the field of computer experiments was done by Sacks et al. [190]. It was formulated from a frequentist approach and the Gaussian process was used to model the residuals of a linear regression. Eventually the idea was translated to a Bayesian modelling perspective in the work of Currin et al. [52] and has been gradually refined over the years. As of today, the paper by Kennedy and O’Hagan [130] offers an introduction to the use of a Gaussian process as an emulator from a Bayesian formulation. The use of Gaussian process surrogates covers a wide range of applications such as uncertainty analysis [170], sensitivity analysis [173], and calibration [113]. In particular, the use of Gaussian process models for reliability analysis [20] are more recent applications.

Considering the success of Gaussian process emulation and their connection to the Bayesian framework for uncertainty quantification, it is a surprise that a single estimate for its hyperparameters is often considered in the literature. The hyperparameters are often fixed to the maximum likelihood estimator (MLE) or the maximum a posteriori (MAP). The reason for this is often accredited to computational convenience. The consequence is that all associated uncertainty in the Gaussian process formulation is summarised in a single candidate. Therefore, the motivation of this dissertation is to integrate a full Bayesian Gaussian process for robust uncertainty quantification in computationally expensive engineering applications. Moreover, computer codes that can run efficiently are not enough to tackle real-world engineering challenges. The trust that can be deposited into these numerical models relies on how well they are calibrated to experimental data. The problem of calibration of expensive computer models given highly constrained experimental settings is addressed by employing history matching [48, 49]. In particular, a full probabilistic perspective is applied to the emulator and propagated when applying history matching. Additionally, the reliability

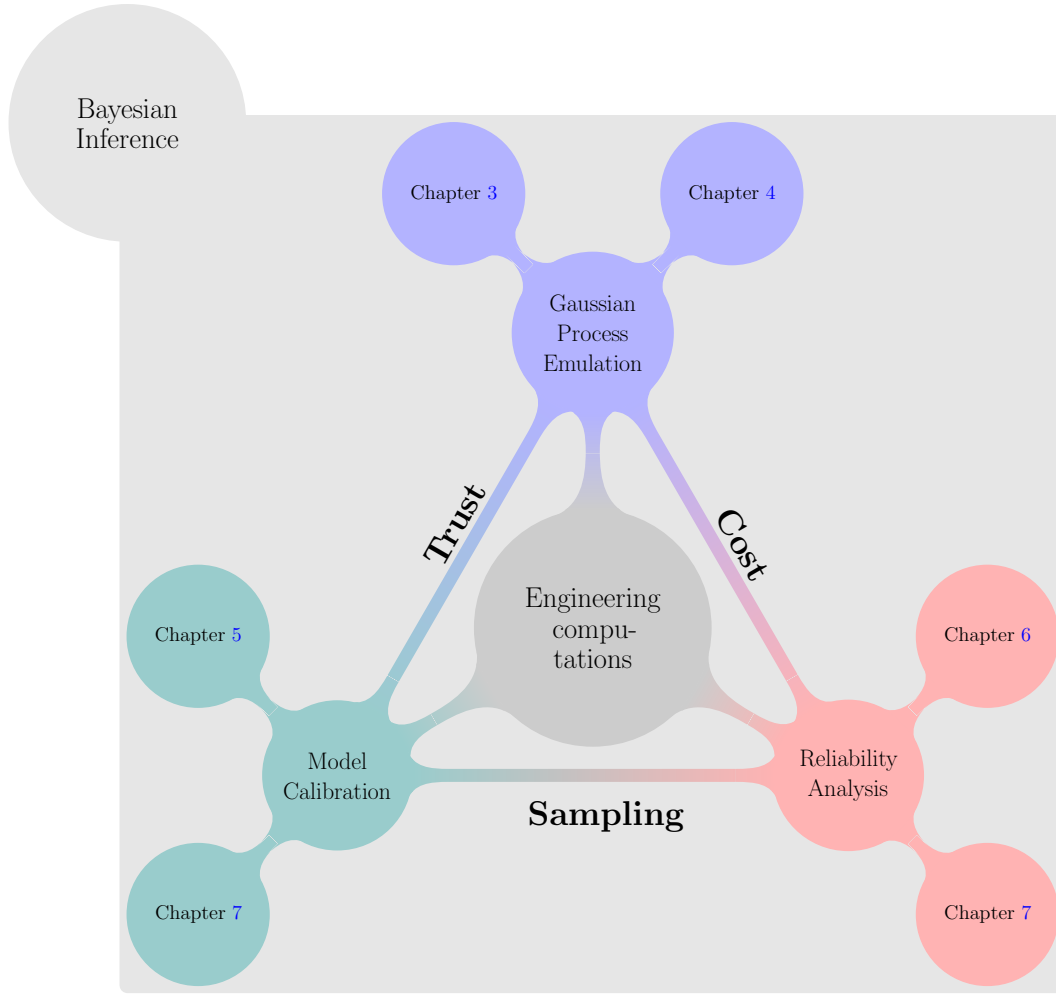
analysis of complex engineering systems is of paramount interest and presents an opportunity of integrating Bayesian inference techniques to improve upon. In particular, techniques such as subset simulation and Bayesian updating with structural reliability methods (BUS) [206] are further developed.

## **1.2 Aims and objectives**

The aim of this dissertation is to develop probabilistic computational algorithms to be applied in Uncertainty Quantification problems of fundamental interest in engineering computations. Specifically, to reduce the cost of complex engineering models to efficiently propagate and quantify uncertainty in the model or its surrogate; calibrate expensive computer codes; and quantify the reliability of complex engineering systems.

The objectives are to implement robust surrogate modelling techniques derived from probabilistic formulations to reduce the computational cost of performing uncertainty quantification; to increase the trust in engineering computer models by developing robust calibration techniques with emphasis on statistical and numerical considerations; to increase the efficiency of the reliability analysis problem in complex engineering systems; to develop calibration algorithms based on a probabilistic framework for implausibility and principles of active learning; to develop a Hamiltonian-based subset simulation technique and incorporate first order information of a system performance function in simulation algorithms; and to develop a connection between reliability analysis and model calibration through a new algorithm that alleviates the need for critical parameters.

The following sections presents an overview of surrogate modelling, history matching and reliability analysis as the foundational driving forces of this dissertation. These ideas and development are summarised in the following semantic map.



## 1.3 Surrogate Modelling

As stated above, surrogate modelling aims to provide fast but accurate approximations to computationally expensive simulators. Let  $f : \mathcal{X} \times \Theta \rightarrow \mathcal{Y}$  denote the surrogate model, where  $\mathcal{X}$  denotes the input space of the simulator,  $\Theta$  denotes the space of parameters controlling the surrogate model, and  $\mathcal{Y} \subseteq \mathbb{R}$  the range of the simulator output. The overall goal of surrogate modelling is to find  $\theta^*$  such that the surrogate  $f(\cdot)$  is as close as possible to the simulator  $\eta(\cdot)$  at every location  $\mathbf{x} \in \mathcal{X}$ . A common approach is to consider the simulator as an unknown function in all its domain except in a selected collection of points. The chosen locations are denoted by  $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  with corresponding outputs  $y_i = \eta(\mathbf{x}_i)$ . The set of input-output pairs is denoted by  $\mathcal{D} = \{(\mathbf{x}_i, y_i) : i = 1, \dots, n\}$  and is referred

to as the set of *training runs*. Most surrogate modelling techniques find  $\boldsymbol{\theta}^*$  by minimising a loss function between observations  $\eta(\mathbf{x}_i)$  and predictions  $f(\mathbf{x}_i, \boldsymbol{\theta})$ . For example, a linear regression would minimise a quadratic loss function by modifying the linear coefficients and taking the training pairs as data. Other approaches have been developed and will be discussed shortly.

For the sake of completeness, some widespread surrogate models are presented in the following subsections. This is by no means an exhaustive list, as this is currently an active research area. For example, learning algorithms from the machine learning community have been adopted as surrogates [see 76, 150, 58], while newer methodologies have been developed [134]. A deeper discussion on surrogate models can be found in [199, Chapter 13].

### 1.3.1 Regression and interpolation-based models

Regression-type models offer the flexibility of treating the simulator  $\eta(\cdot)$  as a *black box*. This means that there is no need to know about the details of the model. The only quantities of interest are the input-output pairs  $\mathcal{D}$ . A surrogate of this type is defined by

$$y_i = f(\mathbf{x}_i, \boldsymbol{\beta}) = \sum_{j=1}^p \beta_j \psi_j(\mathbf{x}_i) + \varepsilon(\mathbf{x}_i), \quad (1.1)$$

where  $\psi_j(\cdot)$  is a collection of known functions, also known as *basis* functions, and  $\varepsilon_i$  is a stochastic error term. The linear expansion of the first term in the right hand side is used to model the *global* trend of the simulator. The second term accounts for unresolved fine-scale behaviour or measurement errors. In regression approaches it is common to assume  $\varepsilon_i = \varepsilon(\mathbf{x}_i)$  as independent and identically distributed (*iid*) random variables with zero mean and finite variance.

#### Quadratic response model

The basis functions can be chosen to accommodate a quadratic response surface. The surrogate model is defined as

$$f(\mathbf{x}, \boldsymbol{\beta}) = \beta_0 + \sum_{j=1}^d \beta_j x_j + \sum_{j=1}^d \beta_{jj} x_j^2 + \sum_{j=1}^d \sum_{k>j}^d \beta_{jk} x_j x_k, \quad (1.2)$$

where  $\boldsymbol{\beta}$  is the vector containing all unknown coefficients in the expansion. The model has  $p = [(d + 1)(d + 2)]/2$  parameters, which can be estimated through least squares. Thus, the required number of simulator runs to enable an accurate estimation of the regression coefficients should be at least  $p = \mathcal{O}(d^2)$ . The quadratic response model is a particular case of a more general family of polynomial expansions. Nonetheless, quadratic models are the most popular among all polynomial expansions of the input variables. However, they can have limited accuracy due to their inability to fine-tune local variations and motivate the use of more flexible models.

### Radial basis functions

This type of surrogate functions employ a more flexible approach than polynomial expansions such as the quadratic model. The surrogate is written as

$$f(\mathbf{x}, \boldsymbol{\beta}) = \sum_{j=1}^n \beta_j \psi_j(\mathbf{x}) + h(\mathbf{x}), \quad (1.3)$$

where  $\psi_j(\mathbf{x}) = \psi(\|\mathbf{x} - \mathbf{x}_j\|)$  are the radial basis functions, and  $h(\cdot)$  denotes the global trend function. The radial basis functions are defined locally by each of the training points. In particular, the basis functions use the Euclidean distance between  $\mathbf{x}$  and the data point  $\mathbf{x}_j$ . Thus, the first term in the right hand side above is designed to capture local variations in the response surface which cannot be captured in  $h$ . A typical formulation uses a constant global trend,  $\beta_0$ . Common choices for the radial function expansion are summarised in Table 1.1.

$\psi(\ \mathbf{x} - \mathbf{x}_j\ )$	Name
$\exp\left(-\frac{\ \mathbf{x} - \mathbf{x}_j\ ^2}{2\theta^2}\right)$	Gaussian
$\ \mathbf{x} - \mathbf{x}_j\ ^\nu$	Power law
$\ \mathbf{x} - \mathbf{x}_j\ ^2 \log(\ \mathbf{x} - \mathbf{x}_j\ )$	Thin plate spline

**Table 1.1:** Examples of radial basis functions for surrogate modelling.

The coefficients  $\boldsymbol{\beta}$  are estimated by imposing an interpolation condition for the training runs. That is,  $\boldsymbol{\beta}$  is estimated subject to the system of  $n$  equalities specified by

$$y_j = f(\mathbf{x}_j, \boldsymbol{\beta}), \quad j = 1, \dots, n, \quad (1.4)$$

along with the constraint

$$\sum_{j=1}^n \beta_j = 0, \quad (1.5)$$

as a consequence of considering a constant global trend. The final prediction can be computed as

$$f(\mathbf{x}, \boldsymbol{\beta}) = \hat{\beta}_0 + k(\mathbf{x})^\top K^{-1}(\mathbf{y} - \hat{\beta}_0 \mathbf{1}), \quad (1.6)$$

where  $\hat{\beta}_0 = (\mathbf{1}^\top K^{-1} \mathbf{1})^{-1} \mathbf{1}^\top K^{-1} \mathbf{y}$ ,  $K \in \mathbb{R}^{n \times n}$  is the Gram matrix of *kernel* evaluations  $K_{ij} = \psi_j(\mathbf{x}_i) = \psi(\|\mathbf{x}_i - \mathbf{x}_j\|)$ , and  $k(\mathbf{x}) \in \mathbb{R}^n$  is the vector of cross evaluations,  $k_i(\mathbf{x}) = \psi(\|\mathbf{x} - \mathbf{x}_i\|)$ . The use of the symbols  $K$  and  $k$  is evocative of the close connection of radial basis functions to kernel methods such as kriging and Gaussian processes. The main difference is that this surrogate does not provide an internal uncertainty measure for the prediction.

### Kriging

This surrogate model is strongly connected to the Gaussian process, which will be discussed in detail as a probabilistic model in Chapter 2 and as a computer code emulator in Chapters 3 and 4. The main difference is that the estimators under Gaussian Processes and kriging stem from different goals. The kriging model is constructed as a surrogate which interpolates observed data. The output of the surrogate  $f(\mathbf{x})$  is assumed to be a random field,

$$f(\mathbf{x}) = m(\mathbf{x})^\top \boldsymbol{\beta} + \varepsilon(\mathbf{x}), \quad (1.7)$$

where  $\varepsilon$  is a zero mean random field with known covariance structure,  $m$  is a known function with output in  $\mathbb{R}^p$  and  $\boldsymbol{\beta}$  is a  $p$ -vector of unknown coefficients. The function  $m(\cdot)$  accommodates for the global trend of the unknown function. The covariance structure is defined by  $\text{cov}(f(\mathbf{x}), f(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}')$ , where  $k(\cdot, \cdot)$  is a known *kernel* function. The covariance function models short term local variations in the predicted surface. Predictions can be computed by using the *best linear unbiased predictor* (BLUP), given the assumed covariance structure. That

is, out of all linear predictors of the form  $\lambda_0 + \boldsymbol{\lambda}^\top \mathbf{y}$ , the BLUP minimises the mean squared error (mse) subject to the unbiasedness constraint  $\mathbb{E}(\lambda_0 + \boldsymbol{\lambda}^\top \mathbf{y}) = \mathbb{E}(f(\mathbf{x}))$  for any vector  $\boldsymbol{\beta}$ . The solution for the constrained optimisation problem is

$$\boldsymbol{\lambda}^\top \mathbf{y} = k^\top K^{-1}(\mathbf{y} - M\hat{\boldsymbol{\beta}}) + m(\mathbf{x})^\top \hat{\boldsymbol{\beta}}, \quad (1.8)$$

where  $\hat{\boldsymbol{\beta}} = (M^\top K^{-1}M)^{-1}M^\top K^{-1}\mathbf{y}$ ,  $M$  is the design matrix obtained after evaluating the mean function  $m(\cdot)$  on the training points, and  $K$  and  $k$  denote the variance-covariance matrix and cross-covariance vector respectively. The mse of the BLUP for an unseen  $\mathbf{x}_0$  is

$$k_0 - k^\top K^{-1}k + \gamma^\top (M^\top K^{-1}M)^{-1}\gamma, \quad (1.9)$$

where  $\gamma = m(\mathbf{x}_0) - M^\top K^{-1}\mathbf{y}$  and  $k_0 = k(\mathbf{x}_0, \mathbf{x}_0)$ .

Common choices for the mean function are the zero function  $m(\cdot) = 0$ , or a constant  $m(\cdot) = 1$ . The former is known as simple kriging; the latter, ordinary kriging. Nonetheless, the flexibility of the model allows for a more general global trend. As suggested by (1.7), it is possible to define a regression-based global trend approach. The full model under this construction is called universal kriging.

As mentioned before, kriging is connected to other models. In particular, an ordinary kriging model assumes  $m(\mathbf{x}) = \beta_0$  for all  $\mathbf{x} \in \mathcal{X}$ . The resulting design matrix can be written as  $M = \mathbf{1}$ . If the radial basis functional form is the same as the kernel function in kriging, both models are equivalent for prediction purposes. That is, the surrogate model prediction under ordinary kriging and the radial basis regression model with constant global trend are the same. The advantage of kriging is that it also incorporates an internal measurement of uncertainty as seen from (1.9).

The kriging model assumes a known covariance structure. This implies that the hyperparameters that appear in the kernel function equation are assumed to be known. However, in practice this is not the case and these parameters need to be estimated from data. The maximum likelihood estimator is usually used with a leave-one-out cross validation scheme to accommodate for small datasets and multiple optima. On the other hand, a Bayesian approach would define a prior distribution for the hyperparameters of the model. This is equivalent to the formulation of a Gaussian process for regression. The main difference is that Gaussian processes define a probabilistic distribution for functions, instead of



building an interpolator as a surrogate. Moreover, the Gaussian process defines a probability distribution on function spaces. Thus, through a Gaussian process random functions can be generated and used as surrogates given observed data.

### Neural networks

Artificial neural networks are at present one of the most well known learning models in machine learning applications. Goodfellow et al. [102] refer to the general field of research concerning neural networks and its variants as *Deep Learning*. The widespread use of deep neural networks in current scientific applications is due to their success in computer vision [47], reinforcement learning [197], speech recognition [21], to name a few. The idea of using a mathematical abstraction for how the brain works can be traced back to the paper by McCulloch and Pitts [151]. The perceptron algorithm by Frank [81] was first used in classification tasks, and was inspired by the analogy to brain models. The most simple neural network can be defined as

$$f(\mathbf{x}, \boldsymbol{\theta}) = \sum_{j=1}^M \alpha_j \Psi(\boldsymbol{\beta}_j^\top \mathbf{x} + \gamma_j), \quad (1.10)$$

where  $\boldsymbol{\theta}$  is the vector of parameters,  $\alpha_j$  is the coefficient associated with neuron  $j$ ,  $\Psi$  is a sigmoidal function with parameters  $\boldsymbol{\beta}$  and  $\gamma_j$ . The general idea is that the input values are combined to generate a response. If this response exceeds a threshold  $\gamma_j$  the neuron emits a signal. The final prediction is made by a linear combination of the signals from the collection of neurons. More complicated architectures can be derived from this simple formulation. Layers of neurons can be added sequentially and their responses are treated as inputs for the next layer of neurons. Thus, the first layer receives the inputs  $\mathbf{x}$ ; the last layer emits a linear combination of the signals from the second-to-last layer to make predictions. The parameters are estimated by minimising an appropriate cost function, usually a quadratic loss. Unfortunately, the cost function is a non convex function and multiple optima can be encountered. Moreover, the number of parameters grows exponentially fast as layers and neurons are added. This has led to study heuristics to avoid overfitting.

Neural networks are known to be universal approximators. That is, any continuous function can be approximated by a neural network. This property makes

them interesting candidates for surrogate modelling. However, they often require big datasets to enable them as robust surrogate models that avoid overfitting. This hinders their applicability in domains of computationally expensive computer codes, where Gaussian processes offer a higher degree of flexibility given scarce amount of data. A more general discussion on different architectures for neural networks and test cases in machine learning applications can be found in [102]. It is important to note that there exists a theoretical connection between Gaussian processes and artificial neural networks. Neal [167] showed that a neural network with a single hidden layer converges to a Gaussian process as the number of neurons increases.

### 1.3.2 High-dimensional model representation

Another type of surrogate modelling techniques consist on variance-based approaches. These type of methods are known as functional ANOVA (Analysis of Variance) in statistics, and high-dimensional model representation (HDMR) or Sobol's representation in the UQ community. This methodology assumes the output from the simulator to be a second order stationary process. That is, the model can be characterized by the first and second statistical moments. In particular, a second order stationary process has constant mean and a covariance function that depends only on the distance of the indices. The HDMR approach considers the randomness of the output a consequence of the randomness in the input variables. The assumption is that input random variables are independent from one another. Thus, the joint distribution of  $\mathcal{X} \subseteq \mathbb{R}^d$  is a product  $d$  marginal distributions. Assuming that the output of the simulator  $\eta(\mathbf{x})$  is square integrable with respect to the joint distribution for  $\mathcal{X}$ , its Sobol decomposition is given by

$$\eta(\mathbf{x}) = f_0 + \sum_{i=1}^d f_i(x_i) + \sum_{1 \leq i < j \leq d} f_{ij}(x_i, x_j) + \cdots + f_{1,\dots,d}(\mathbf{x}), \quad (1.11)$$

where  $f_0$  represents the mean response of  $\eta(\cdot)$ ,  $f_i$  the individual contributions of each variable,  $f_{ij}$  the interaction and contribution from pairs of variables  $i$  and  $j$ , and so on. In general, this type of methods have proven to be effective for sensitivity analysis. That is, to identify the most influential variables in the variance of a simulator.

### Polynomial chaos expansions

A surrogate model for  $\eta(\cdot)$  belongs to the class of polynomial chaos expansions if it is written as

$$f(\mathbf{x}, \boldsymbol{\theta}) = \sum_{\alpha \in \mathcal{A}} \theta_{\alpha} \Psi_{\alpha}(\mathbf{x}), \quad (1.12)$$

where  $\mathcal{A}$  is a set of multi-indices  $\alpha = (\alpha_1, \dots, \alpha_d)$ ,  $\{\Psi_{\alpha}\}$  is a set of multivariate family of polynomials orthogonal with respect to the inner product defined by the density of  $\mathcal{X}$ , and  $\theta_{\alpha}$  are the unknown coefficients associated with each polynomial. Each of the multivariate polynomials  $\Psi_{\alpha}$  can be obtained as a tensor product of univariate polynomials

$$\Psi_{\alpha}(\mathbf{x}) = \psi_{\alpha_1}^{(1)}(x_1) \times \dots \times \psi_{\alpha_n}^{(n)}(x_n), \quad (1.13)$$

where  $\psi_{\alpha_j}^{(j)}$  denotes the  $\alpha$ -th order polynomial for the  $j$ -variable. Note that the orthogonality condition on the family of polynomials makes the expansion completely dependent on the assumed joint distribution for the input variables. For example, if the assumed probability distribution is a Gaussian, then the family of polynomials used is the Hermite polynomials. This assumption plays an important role in the methods used to find the unknown coefficients. For example, quadrature rules actively exploit the orthogonal property of the polynomial basis by approximating the integrals associated to the inner product. Quadrature points are chosen based on the assumed probabilistic model and these points define the set of configuration points to run the expensive simulator. This strategy is known as a non-intrusive method for solving the polynomial chaos expansion. Note that this renders the estimation of the polynomial expansion coefficients, and more importantly the design of simulator runs, completely dependent on the assumed prior distribution of simulator inputs. See [174] for a detailed discussion on polynomial chaos and non-intrusive methods.

The formulation of the surrogate in (1.12) is an infinite series in the space of polynomials, which in turn, is an infinite dimensional Hilbert space. For computational convenience, it is common to truncate the series and keep only the polynomials of degree up to  $p$ . This means that the total number of unknown coefficients for the expansion is

$$\frac{(d+p)!}{d! p!}. \quad (1.14)$$

Thus the number of coefficients to be estimated grows exponentially fast as the the number of dimensions in the simulator's input increases. Several regression-based approaches to overcome this limitation have been proposed in [33, 34]. Nonetheless, the main drawback when compared to the Gaussian process model is still the computational burden of running the simulator to generate a training dataset. The reason is that the assumptions on the prior distribution of the simulator's input determines the training dataset to be used to learn the polynomial chaos. Such is the case of using a non-intrusive method based on quadrature rules. Alternatively, a regression-based approach does not guarantee that the chosen polynomial family is a good mathematical object to approximate the simulator output. A more thorough discussion on Polynomial chaos can be found in [93, 94]. A summary and a comparison of Polynomial chaos expansions and Gaussian processes as computer code emulators can be found in [175]. O'Hagan [174] provides a thorough discussion on polynomial chaos surrogates from a statistician perspective.

### Low rank tensor approximations

Low rank tensor approximations have been recently developed as an alternative approach to polynomial chaos expansions [134]. In particular, they have been used in sensitivity and reliability analysis [136, 135]. The simulator is represented as a sum of a finite number of rank-one functions. That is, the surrogate is referred as a canonical decomposition with rank equal to the number of rank-one components in the model. A rank-one function is written as

$$\omega_l(\mathbf{x}) = \prod_{i=1}^d v_l^{(i)}(x_i), \quad (1.15)$$

where  $v_l^{(i)}$  denotes a univariate function of  $\mathbf{x}_i$ . The surrogate model is defined as

$$f(\mathbf{x}, \boldsymbol{\theta}) = \sum_{l=1}^R \theta_l \omega_l(\mathbf{x}), \quad (1.16)$$

where  $R$  is the rank of the approximation and  $\boldsymbol{\theta}$  is a vector of unknown coefficients.

Lastly, the family of univariate functions  $v_l^{(i)}$  is chosen as an orthonormal basis of polynomials with respect with the marginal distributions for  $x_i$ . This implies an expansion on each univariate function as

$$v_l^{(i)}(x_i) = \sum_{k=0}^{p_i} z_{k,l}^{(i)} P_k^{(i)}(x_i), \quad (1.17)$$

where  $P_k^{(i)}$  is the  $k$ -th degree polynomial in the  $i$ -th input,  $p_i$  is the maximum degree of  $P_k^{(i)}$  and  $z_{k,l}^{(i)}$  additional unknown coefficients for each univariate function. The total number of unknown parameters to be estimated is

$$R \sum_{i=1}^d (p_i + 1), \quad (1.18)$$

which scales *linearly* with the number of dimensions. This provides an advantage with respect to polynomial chaos expansions. Details on the estimation of the parameters for a low rank tensor approximation can be found in [136].

## 1.4 History Matching

So far in our discussion, a simulator has been defined as a computational implementation of a mathematical model. This allows one to describe and study a physical process of interest through computer code evaluations. These simulators have been introduced as mappings  $\eta(\cdot)$  of a certain input domain  $\mathcal{X}$  to a range of possible values  $\mathcal{Y}$ . The input domain considers vectors of inputs  $\mathbf{x}$ , where some components are observed physical parameters in the realisation of physical process, some are inherent to the mathematical model, and some others are related to the computational code being used. These parameters allow the simulator to approximate the process it is modelling. Thus, if a simulator is expected to be used as a reliable approximation of reality, the code should be well calibrated. That means that an input configuration  $\mathbf{x}^*$  is known and it produces the closest output to observed data. Typically,  $\mathbf{x}^*$  is found by both extensive experimentation and domain knowledge.

The problem is that modern simulators are often computationally expensive which means that thorough exploration of the input space is not feasible. This motivates the use of surrogate models, like the ones presented in the previous

section. The objective is to use fast but accurate approximations to the simulator to explore more efficiently the input space. In practice, this exploration means to minimise a cost function relating the computer code output, now being approximated by the surrogate model, and a collection of observed data. This strategy allows one to find  $\mathbf{x}^*$  to use a well-calibrated simulator.

In general, there are many uncertainties associated with the previous methodology. As discussed previously, uncertainty can arise from the mathematical model, the numerical implementation and the measurement process. A Bayesian approach would treat  $\mathbf{x}^*$  as a true but unknown parameter, and a posterior distribution for  $\mathbf{x}^*$  would be the result of updating the prior specification given collected data. Nonetheless, there might not be enough measured data to believe that there is a unique choice for  $\mathbf{x}^*$ . Actually, to choose a single point might not be possible at all. In this setting, *history matching* allows the identification of collections of simulator evaluations which are consistent with measured data, within the levels of uncertainty associated with the problem. Thus, modelling the measurement error, the model discrepancy and the code uncertainty derived from the surrogate is paramount in history matching applications [99].

The identification of a single candidate  $\mathbf{x}^*$  is relaxed for history matching. This relaxation offers a change of goal which is an advantage of history matching. The surrogate is used to explore the input space in order to find regions on which the simulator gives acceptable matches to the data. History matching allows one to discard regions of input space even in cases where the simulator has a multi-dimensional output, as in [216]. This reduction of input space is done with the hope of actually reducing the search space on which other traditional approaches such as Bayesian calibration is easier to handle. In particular, a probabilistic version of history matching has been developed under the umbrella of *Bayes linear analysis* [48]. This framework considers expectation as the foundational component of probabilistic inference. The linear properties of expectation as an operator are inherited and thus the term linear Bayes. A thorough treatment of probability through this approach can be found in the work of DeFinetti [54]. The main advantage is that a minimal set of assumptions from a subjective point of view can be defined. This implies that only the first and second moments are specified, instead of complete distributional assumptions. In the linear Bayes setting, probabilistic updating is performed under the notion of Bayes linear adjustment [see 100, for a thorough exposition]. That is,  $\mathbb{E}_z(y)$  and  $\mathbb{V}_z(y)$  are the

expectation and variance of the vector  $y$  adjusted by the observations  $z$ . These quantities are given by

$$\mathbb{E}_z(y) = \mathbb{E}(y) + \text{Cov}(y, z) \mathbb{V}(z)^{-1}(z - \mathbb{E}(z)) \quad (1.19)$$

$$\mathbb{V}_z(y) = \mathbb{V}(y) - \text{Cov}(y, z) \mathbb{V}(z)^{-1} \text{Cov}(z, y). \quad (1.20)$$

This updating mechanism allows one to specify expectation and variances of untested input configurations of the simulator. This allows one to discard regions of parameter space by pointwise evaluations of the expected code output and the variance term which includes all modelled uncertainties. In particular, history matching assesses the number of standard deviations between the observed data point  $z$  and the expected surrogate for  $\eta(\mathbf{x})$ . The number of standard deviations can be computed as

$$I(x) = \frac{|z - \mathbb{E}(f(\mathbf{x}))|}{\sqrt{\mathbb{V}(\mathbf{x}) + \sigma_{\text{me}}^2(\mathbf{x}) + \sigma_{\text{md}}^2(\mathbf{x})}}, \quad (1.21)$$

where  $\sigma_{\text{me}}^2(\mathbf{x})$  and  $\sigma_{\text{md}}^2(\mathbf{x})$  denote the assessed measurement error and model discrepancy respectively (possibly heteroskedastic), and  $I(\cdot)$  denotes the *implausibility measure*. A threshold of three standard deviations is commonly used as at least 95% of probability mass is contained within three standard deviations in unimodal continuous distributions. This is known as Pukelsheim's three sigma rule [182]. Thus, a point is deemed implausible if  $I(\mathbf{x})$  exceeds the threshold 3, written as  $I(\mathbf{x}) > 3$ , and is consequently discarded of the analysis.

The history matching framework sequentially removes regions of parameter space using the implausibility function. The input regions that are considered “non-implausible” are kept and are sampled to refocus the emulator for the next stage and further reduce the non-implausible region. The procedure of resampling, re-emulating, and reducing the non-implausible space can be done until the space is not reduced further. Each iteration of this procedure is known as a *wave* [216]. This results in regions of parameter space where points  $\mathbf{x}$  would give acceptable fits to historical data. A more thorough discussion on history matching can be found in the original papers [48] and [49]. A more recent discussion on history matching can be found in [99]. Additionally, in Chapter 5 history matching will be developed further by proposing a probabilistic version of the implausibility function. This is due to the probabilistic nature of a Gaussian process as a

computer code emulator under a full Bayesian formulation.

## 1.5 Reliability Analysis

Another topic that is essential in uncertainty quantification is that of reliability analysis (RA). Its main focus of interest is the estimation of probabilities for rare events. A rare event can be defined as the realisation of a state of nature with an almost negligible probability. In engineering systems and applications the realisation of such events generate catastrophic consequences. Thus, an adequate estimation of the probability of such type of outcomes is paramount to communicate and perform uncertainty analysis.

A rare event is often associated with the failure of a certain system. Failure is characterised as the exceedance of a critical response over a given threshold. The response of a system can be represented mathematically as a function  $h(\mathbf{x})$ . It is defined in an input domain  $\mathcal{X} \subseteq \mathbb{R}^d$  which encompasses possible configurations of the system. The failure domain is defined as

$$F = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) > b\}, \quad (1.22)$$

where  $b$  denotes the critical threshold level.

The system configuration is subject to variations which can be represented as a probability density function  $\pi(\mathbf{x})$ . As before, this is done to express uncertainty in the system's configuration. Thus, letting  $P_F$  denote the probability of failure of the system, it can be computed using

$$P_F = \int_{\mathcal{X}} \mathbf{1}(\mathbf{x} \in F) \pi(\mathbf{x}) d\mathbf{x}, \quad (1.23)$$

where  $\mathbf{1}(\mathbf{x} \in F)$  denotes the indicator function which is equal to 1 if  $\mathbf{x} \in F$ , and 0 otherwise. The main challenge is that the integral cannot be computed in closed form and efficient approximation techniques are needed. A naive implementation would approximate the integral with a Monte Carlo estimate using random samples generated from the density  $\pi(\mathbf{x})$ . However, the relative error of the estimation can only be decreased if an enormous amount of simulations is performed. For example, if the probability of failure is of order  $10^{-k}$ , the number of simulations needed to keep a relative 10% deviation is of order  $10^{k+2}$  [12].



The estimation of the probability of failure by means of the Monte Carlo method presents a concern if the response function is a computationally expensive computer model. This estimation would require a massive use of computational resources or would take too long to be useful. In this setting, *subset simulation* alleviates the computational demands by generating samples in  $F$  by means of a decreasing sequence of  $M$  intermediate failure events, denoted as  $F = F_M \subset \cdots \subset F_0 = \mathcal{X}$ . Thus, the probability of failure can be estimated with a lower computational cost as the product

$$P_F = \pi(F_M | F_{M-1}) \times \cdots \times \pi(F_1 | F_0) \times \pi(F_0), \quad (1.24)$$

where each  $\pi(F_{i+1} | F_i)$  denotes the intermediate probabilities of failure. A thorough discussion on how to define the intermediate levels can be consulted in the original paper [9].

## 1.6 Areas of Opportunity

The previous discussion has presented several topics of interest in Uncertainty Quantification to be developed further in this dissertation. In particular, the use of probabilistic surrogate models (emulators) in engineering systems where computationally expensive simulators need to be exploited for appropriate quantification of modelled uncertainties. In the literature, a Bayesian framework for surrogate modelling is often adopted, although to a certain extent. This provides opportunities of exploring, for example, a full Bayesian characterisation of surrogate models. Thus, the following areas of opportunity have been identified based on the following classification.

### 1. Computer code emulation

- a) Exploit a full Bayesian characterisation of Gaussian processes for surrogate modelling of computationally expensive simulators.
- b) Develop effective sampling algorithms that can cope with limited amounts of data in surrogate modelling, and avoid traps of isolated local maxima in posterior distributions.
- c) Assess the robustness of quantified uncertainties by means of a Gaussian process with sampled hyperparameters instead of a typical maximisation

approach.

2. History matching

- a) Incorporate a full probabilistic characterisation of the surrogate model for history matching.
- b) Adopt efficient sampling techniques to identify and explore the non-implausible domains in history matching.
- c) Provide a study of learning functions to guide and refocus the emulator used at each wave in history matching.

3. Reliability analysis

- a) Exploit the direct connection between reliability analysis and history matching due to the failure domain and the non-implausible characterisation.
- b) Develop efficient sampling algorithms for constrained domains as found in reliability analysis and history matching.
- c) Adapt existing probabilistic programming software developed for machine learning application to be used in general engineering applications.

## 1.7 Layout of the Dissertation

This dissertation is organised as follows. In Chapter 2, a brief overview on Bayesian inference and Gaussian processes as regression models is provided. In Chapter 3, Gaussian processes are used as computer code emulators and their implementation is discussed. Additionally, two newly developed sampling algorithms are presented and studied in the context of small datasets and multimodal posterior distributions in Chapters 3 and 4. In Chapter 5, history matching is presented as a calibration technique in computationally restricted settings. In particular, Gaussian processes are used as emulators in history matching as surrogates for computationally expensive simulators. Moreover, the sampling algorithms developed in Chapters 3 and 4 are used in this context to be able to sample from the non-implausible regions. Three active learning criteria are discussed to further improve the emulator in each wave of history matching. In Chapter 6, a Hamiltonian-based subset simulation algorithm is developed in order to improve the efficiency of

sampling in reliability analysis. This is because the use of implausibility functions in history matching resemble the formulation of failure domains in reliability analysis. Furthermore, Chapter 7 presents an extension of subset simulation to more general Bayesian inference problems. In particular, a reformulation of the BUS (Bayesian Updating with Structural reliability-based sampling) algorithm is presented which circumvents the definition of a critical parameter. Finally, Chapter 8 provides conclusions and suggests future directions of research.



## CHAPTER 2

---

### Preliminaries

---

This chapter presents the theoretical foundations for this dissertation. The first half of the chapter presents an overview of Bayesian inference as a theory that characterises coherent decision-makers facing uncertainty. Bayesian theory treats probability as a formal representation of the decision-maker's degree of belief in the realisation of an uncertain event. Within the framework of this dissertation, uncertainty arises due to the high computational demands of complex simulators. Surrogate models are used to overcome this computational burden. Thus, the objective is to build an educated guess of the response surface for a given computer model with an appropriate input parameter space. In particular, Gaussian processes build reliable surrogates in light of limited data. Furthermore, they provide an internal measure of uncertainty associated with its own predictions. As such, Gaussian processes allow one to quantify and update uncertainty in infinite dimensional function spaces. Thus, the second half of the chapter presents a brief overview on the formulation of a Gaussian process as a regression technique. Some insight in the implicit assumptions being made by the Gaussian process model is presented as well. The use of Gaussian process as computer code surrogates is introduced in Chapter 3 where sampling algorithms are developed for this particular Bayesian model.

## 2.1 Bayesian Inference

Laplace wrote in 1814 that “Probability theory is at bottom nothing but common sense reduced to calculus” [140]. In modern mathematics, Measure theory provides the axiomatic framework to manipulate probability theory. However, at the very core, there are different schools of thought regarding the notion of what probability means. The *objective* school of thought postulates that the probability of the occurrence of an event, is a property purely attributed to the event in question, Feller [refer to 78]. Thus, the probability of an uncertain event can be computed and estimated through a limiting sequence of frequencies. The so-called ratio of counts of the realisation of the outcome of interest, against the counts of all possibilities. In contrast, the *subjective* school of thought treats probability as the decision-maker’s degree of belief on the realisation of the event of interest. In particular, probability is a subjective quantity attributed to the decision-maker when dealing with uncertain events. As a result, probability follows from the decision-maker’s strategy to maximise her utility satisfying certain rational principles. Hence, probabilities can be elicited as the decision-maker’s willingness to bet on the realisation of an uncertain event. Furthermore, these degree of beliefs can be updated by means of a simple mathematical rule known as Bayes’ theorem. This result is attributed to Rev. Thomas Bayes [27]. A proper treatment of the foundations of probability from a subjective view can be found in the work of Savage [194] and DeFinetti [54] (originally published in 1974).

In the context of Uncertainty Quantification, Bayesian inference provides a solid mathematical foundation to operate and update probability models in the evidence of data. In Bernardo’s words “Bayesian statistics offers a rationalist theory of personalistic beliefs in contexts of uncertainty, with the central aim of characterising how an individual should act in order to avoid certain kinds of undesirable behavioural inconsistencies” [27]. As such, this dissertation follows the Bayesian framework to quantify uncertainty in engineering applications. In the following paragraphs the general procedure for updating prior beliefs is presented.

Let  $\mathcal{D} = \{x_1, \dots, x_n\}$  denote  $n$  independent realisations of an observable quantity. Let  $p(x|\boldsymbol{\theta})$  denote the assumed probabilistic model that generated such data. That is, each  $x_i$  is an independent realisation from the model  $p(x|\boldsymbol{\theta})$ , also known as the *likelihood*. The joint likelihood function, which follows from the assumption of independent realisations of each datum, can be written as  $p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{i=1}^n p(x_i|\boldsymbol{\theta})$ .

The parameter  $\theta$  represents an unobservable quantity as it is only a mathematical construct in the model. The goal is to make inferences on the value of such a parameter to understand better the data-generation mechanism. There is no direct observation of  $\theta$ , but the analyst can state her prior beliefs on such quantities by a probabilistic model  $p(\theta)$ . The latter is known as the *prior* distribution. Together with Bayes' theorem, the analyst updates her beliefs based on the following mathematical rule

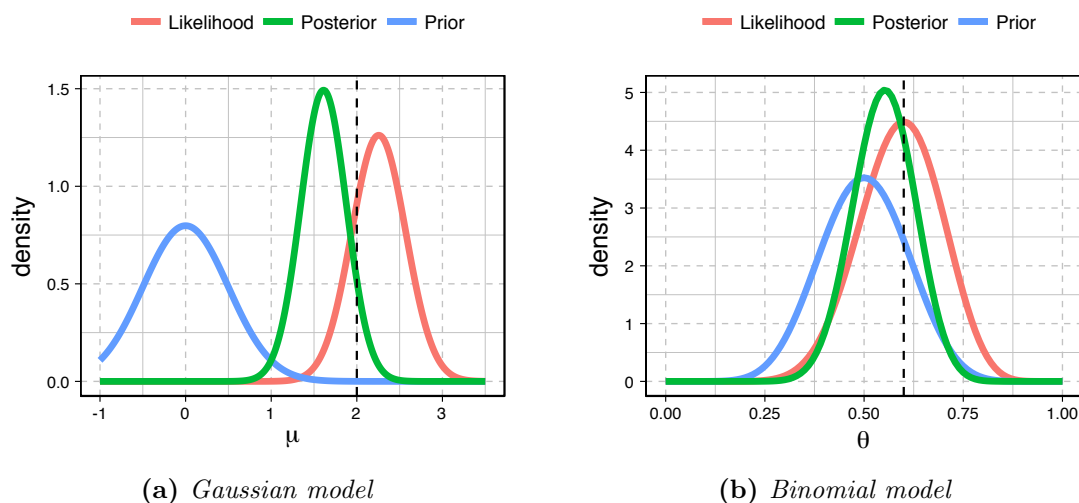
$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}, \quad (2.1)$$

where  $p(\theta|\mathcal{D})$  denotes the so-called *posterior* distribution and  $p(\mathcal{D})$  denotes the *evidence* of the data. The posterior distribution represents the updated beliefs of the analyst after mixing the likelihood, the prior and the data. The evidence is the normalising factor of the posterior distribution and is calculated by marginalising the unobserved quantities. Assuming  $\theta$  is a continuous parameter, the evidence is computed as

$$p(\mathcal{D}) = \int_{\Theta} p(\mathcal{D}|\theta)p(\theta)d\theta, \quad (2.2)$$

where  $\Theta$  denotes the support of  $\theta$ . That is, the set of all possible realisations of  $\theta$ .

Figure 2.1 depicts two examples of Bayesian updating. The left panel depicts inference of a Gaussian model with unknown mean parameter denoted by  $\mu$ . In this case, the variance  $\sigma^2$  is assumed to be known. A sample of independent and identically distributed (*iid*) realisations is collected. In this case, the likelihood is a Gaussian distribution and together with a Gaussian prior leads to a Gaussian posterior for  $\theta$ . Note that the posterior provides a balance between the prior and likelihood functions. The likelihood model dominates the prior and gives coverage to the true parameter value, shown as vertical dashed line. The right panel, shows the Bayesian updating for  $n$  *iid* Bernoulli experiments. The likelihood model is a Binomial distribution with unknown probability of success, denoted by  $\theta$ . In this case, the likelihood is maximised in the true value of  $\theta$ , shown as vertical dashed line. However, under a symmetric Beta(10, 10) prior, the posterior produces a more conservative mode.



**Figure 2.1:** Bayesian inference examples. On the left, a Gaussian model with unknown location parameter  $\mu$ , and known variance  $\sigma^2$ . On the right, A Binomial model with unknown probability of success  $\theta$ . The three curves are shown to illustrate the interplay of prior and likelihood assumptions to yield the posterior distribution.

### 2.1.1 Prior distributions

The choice of an appropriate prior distribution has remained an open question in research. The alternatives have ranged from priors that have the least effect on the likelihood to highly informative priors. Priors with little effect on the likelihood are known as *noninformative*. The name reflects that no additional prior information on the model's parameters is known. Informative priors arise from situations where there is some evidence on possible values taken by the parameters, *e.g.* appropriate scale or range of values. This information usually comes from domain knowledge and it is updated through the use of the Bayesian framework.

Noninformative prior specification has been motivated on scenarios where objective inference is desired. This setting aims to let the data to speak for itself without any inferential bias. In this sense, Jeffreys' prior is typically used for one parameter models [122]. However, it has been found to be inappropriate when dealing with multiparameter families [25]. The use of *reference* priors allows one to bypass this limitation [24]. However, they are model dependent and difficult to derive for complex models. Alternatively, *weakly informative* priors lie between



the noninformative and highly informative prior specification. This type of prior distribution aims to penalise the complexity of the model and have been found useful in hierarchical models [89, 198]. In summary, the analyst is free to choose whatever prior distribution reflects her current state of belief. However, it has been pointed out by Gelman et al. [91], that this specification should be consistent with the model as a data-generative mechanism.

### 2.1.2 Prediction

Having specified the complete model, the analyst is expected to report results from her posterior beliefs. This can be done, for example, by communicating summaries of the posterior distribution. One is likely to opt for the most probable explanation. That is, to report back the Maximum a Posteriori (MAP) estimate of  $\boldsymbol{\theta}$ . Although other options include credible intervals or other statistics. The MAP is a simple solution as it involves the optimisation of the numerator in (2.1), effectively avoiding the computation of the evidence in (2.2). This is desirable as the integral involved in the calculation of the normalising constant does not have, in general, a closed-form solution. However, more often than not, the objective of the study incorporates performing predictions as well. Thus, another layer of complexity in the analysis arises. In this situations, the MAP estimate for the parameters might provide a short-sighted summary. Let  $x^*$  denote the value to be predicted from the probability model. The task now involves using all data available and the analyst's model assumptions for predictions. For this, the *predictive posterior* distribution is computed as

$$p(x^*|\mathcal{D}) = \int_{\boldsymbol{\Theta}} p(x^*|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta}, \quad (2.3)$$

which follows from the independence of the observables given the model's parameters. The predicted posterior effectively incorporates both data and modelling assumptions through the posterior distribution. However, an intractable integral needs to be solved to make predictions. This motivates the development of Markov Chain Monte Carlo (MCMC).

### 2.1.3 Reporting inferences and Markov Chain Monte Carlo

The MAP estimate could be used to approximate the integral, but this strategy does not incorporate the uncertainty of  $\theta$ . This suggests that more information can be extracted from the probabilistic formulation of the full Bayesian model. The development of Markov Chain Monte Carlo (MCMC) has provided a way to report results from the posterior in a systematic way [92, 188]. The success of MCMC methods stems from both the overall improvement of modern computing power and the acknowledgement of shortcomings in high dimensional settings. An example of the latter is the so-called concentration of measure. That is, the non-intuitive phenomenon where neighbourhoods around the MAP estimate do not contain the most of the probability mass of the model [127].

In terms of computational implementation, *Probabilistic programming* allows one to write the model and prior beliefs as a computer code. By feeding data to this model and using automatic sampling algorithms, one is able to generate more informative reports from the posterior distribution using MCMC. Common choices to perform this calculations are probabilistic programming software like JAGS [118], Stan [40], PyMC3 [192] and Edward [213].

The computation of the integral in (2.3) is approximated by an average if samples of the posterior distribution can be generated exactly. This is called a Monte Carlo approximation. The quality does not depend on the dimensionality of the problem, but in the number of samples generated. This result is guaranteed by both the Strong Law of Large Numbers and the Central Limit Theorem. The former states that sample averages converge almost surely to the theoretical expectation. The latter states an asymptotic Gaussian distribution centred in the theoretical expectation. The variance reduces as the number of generated samples increase with rate  $1/\sqrt{n}$ , where  $n$  denotes the number of samples. However, for more general model specifications it is not possible to generate correct independent random numbers. In practice only a handful of simple cases are possible [see part V and VI of 40, for a list of such distributions].

In the general case, where no such random number generators exist, the strategy is to settle for a sequence of correlated random numbers which distribution eventually converges to the distribution of interest. This is achieved by a sequential generation of random numbers that satisfy the Markov property. Let  $\{\theta^{(k)}\}_{k=1}^m$  denote a sequence of random numbers. It is said that  $\{\theta^{(k)}\}_{k=1}^m$  follows the

Markov property if the equality  $p(\boldsymbol{\theta}^{(k+1)}|\boldsymbol{\theta}^{(k)}, \dots, \boldsymbol{\theta}^{(0)}) = p(\boldsymbol{\theta}^{(k+1)}|\boldsymbol{\theta}^{(k)})$  is satisfied. Starting from the current state  $\boldsymbol{\theta}$ , a new element for the chain is set to  $\boldsymbol{\theta}^*$ , if it is accepted with probability

$$a(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \min \left\{ 1, \frac{p(\boldsymbol{\theta}|\boldsymbol{\theta}^*) \pi(\boldsymbol{\theta}^*|\mathcal{D})}{p(\boldsymbol{\theta}^*|\boldsymbol{\theta}) \pi(\boldsymbol{\theta}|\mathcal{D})} \right\}, \quad (2.4)$$

where  $p(\boldsymbol{\theta}^*|\boldsymbol{\theta})$  denotes the transition operator. If the sample is not accepted, the chain is grown with a copy of  $\boldsymbol{\theta}$ . The transition operator is a random number generator that proposes a new state for the Markov chain. Common choices are symmetric distributions as the Gaussian or a Uniform. The original implementation of this idea is known as the Metropolis algorithm [153, 154]. The use of more general transition operators, not necessarily symmetric, was provided by Hastings and is known as the Metropolis-Hasting algorithm [111]. The ratio in (2.4) accounts for this more general setting and is known as the Metropolis-Hasting ratio. If the Markov Chain is grown with transitions governed by (2.4), the distribution of the samples is guaranteed to eventually satisfy the ergodic property. That is, after a *burn in* period, the samples generated by this mechanism will follow the target distribution  $p(\boldsymbol{\theta}|\mathcal{D})$ . The rate of convergence depends on the ability to generate good candidates, although a high number of samples is usually recommended. Several diagnostics for MCMC convergence are available and can be consulted in [188] and implemented in the coda software in R [180]. A more thorough introduction on MCMC and on convergence of MCMC methods can be found in [92]. Several modifications to the Metropolis Hastings algorithm have been proposed in the literature and can be found in [37], [189], [188] and [143]. The sampling algorithms developed in this dissertation can be thought as samplers from the family of Sequential Monte Carlo (SMC) methods [67]. SMC methods have gained popularity in recent applications due to their flexibility and success cases. See [67] for a thorough exposition on theoretical and practical developments of SMC methods. In particular, SMC methods are able to exploit modern computational architectures and address the sequential nature of probability updating of Bayesian inference applications [66, 104].

## 2.2 Gaussian Processes

At an abstract level, Gaussian processes can be thought as a probability distribution over functions. In this section, the theory is presented in the context of nonparametric regression. Firstly, a linear regression model is introduced, linear regression model is a statistical technique designed to make predictions about a quantity of interest,  $y$ , from a linear combination of certain observable quantities  $\mathbf{x} \in \mathbb{R}^p$ . In this setting, let  $\mathcal{D}$  denote the collection of  $n$  input-output observations. That is,  $\mathcal{D} = \{(\mathbf{x}_i, y_i) : i = 1, \dots, n\}$  is a *dataset* of  $n$  observation pairs, consisting of the target  $y$  and predictors  $\mathbf{x}$ . Given the data, a basic linear model would assume the relationship

$$y \approx \beta_1 x_1 + \dots + \beta_d x_d, \quad (2.5)$$

where  $\boldsymbol{\beta}$  is a vector of unknown coefficients to be estimated by minimising a loss function  $\mathcal{L}$ . Typically, the loss function used is a quadratic function which aims to penalise deviations from the true value in  $y$  written as

$$\mathcal{L}(\mathcal{D}, \boldsymbol{\beta}) = \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2. \quad (2.6)$$

In statistical language, this is interpreted as assuming a Gaussian model for  $y$  with mean  $\mathbf{x}^\top \boldsymbol{\beta}$  and common standard deviation denoted by  $\sigma$ . The solution for  $\boldsymbol{\beta}$  is computed by minimising the log-likelihood function, or equivalently the quadratic loss function written above.

The linear regression model captures the linear relationship from the observed predictors with the outcome of interest. As restrictive as it is, it can also accommodate for different extensions, as discussed in Chapter 1. In particular, the linear model (2.5) can readily be extended by two approaches. The first approach would consider a family of known basis functions  $\phi_j(\mathbf{x})$  to be used as predictors in the linear expansion. This is the approach taken by radial basis functions or polynomial models [32, 83]. The second, a probabilistic approach, considers a prior distribution on every possible function that could be used as a mapping from predictors to target variable.

The first approach has an obvious problem as there is no guarantee that the considered family of functions is appropriate. An inflexible class would lead to

poor predictions, while a richer class could lead to an overfitted model. On the other hand, the general problem of setting a prior distribution on functions seems like an unsolvable problem. Gaussian processes alleviate these concerns by generalising the multivariate Gaussian distribution to infinite dimensional spaces. In contrast to assigning probabilities to scalars or vectors in an Euclidean space, Gaussian processes govern the properties of the allowed functions in probability space.

### 2.2.1 Priors on random linear functions

In this section a simple construction of random functions is presented. This strategy builds up from the linear model (2.5). Diaconis [62] traces this construction all the way back to Poincaré [181]. The strategy is to consider the coefficients  $\beta$  in the linear model as Gaussian random variables,  $\beta \sim \mathcal{N}(\beta_0, \Sigma_0)$ . Without loss of generality, it is assumed that  $\beta_0 = 0$ . Thus,  $\beta$  is seen as a zero mean random vector with possibly correlated components.

The model is constructed as follows. Under the assumption of independent observations, and given the vectors of coefficients and predictors, the target variable is assumed to be distributed as

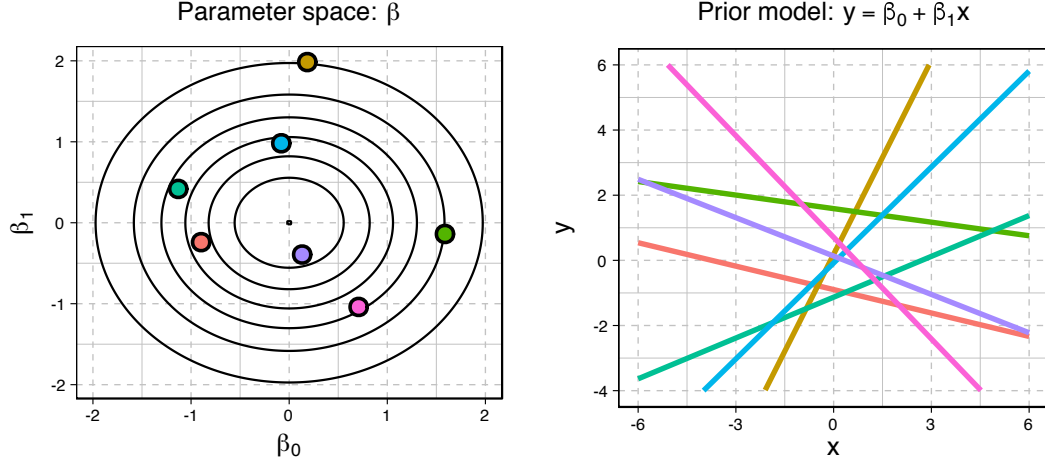
$$p(\mathbf{y}|X, \beta) = \mathcal{N}(X\beta, \sigma^2 I_n), \quad (2.7)$$

where as before, data comes in pairs  $(\mathbf{x}_i, y_i)$ . The vector  $\mathbf{y} \in \mathbb{R}^n$  contains all recorded target values,  $X \in \mathbb{R}^{n \times d}$  is the matrix formed by placing in each row every observed vector corresponding to predictors, and  $I_n \in \mathbb{R}^{n \times n}$  denotes the identity matrix. Note that every realization of  $\beta$  from the prior, defines a different linear model as depicted in Figure 2.2. In this sense, the prior distribution over  $\beta$  reflects our belief on which linear models are deemed possible.

The posterior distribution is computed as

$$\begin{aligned} p(\beta|X, \mathbf{y}) &= \frac{p(\mathbf{y}|X, \beta) p(\beta)}{p(\mathbf{y}|X)} \\ &\propto p(\mathbf{y}|X, \beta) p(\beta), \end{aligned} \quad (2.8)$$

where the normalising constant is discarded for computational convenience. The reason behind this is that the evidence is a function of the available data, and the



**Figure 2.2:** *Prior assumptions on a linear model. The contour levels of the prior distribution assumed for the coefficients are shown. Random samples are taken from the joint distribution of  $\beta = (\beta_0, \beta_1)$ , and are depicted as dots. Each realisation of the prior induces a linear model as shown on the right with matching colours. Note that the prior does not show preference for a particular linear model.*

focus is on the distribution for  $\beta$ . The Gaussian assumption on both the target and the coefficients leads to the Gaussian posterior

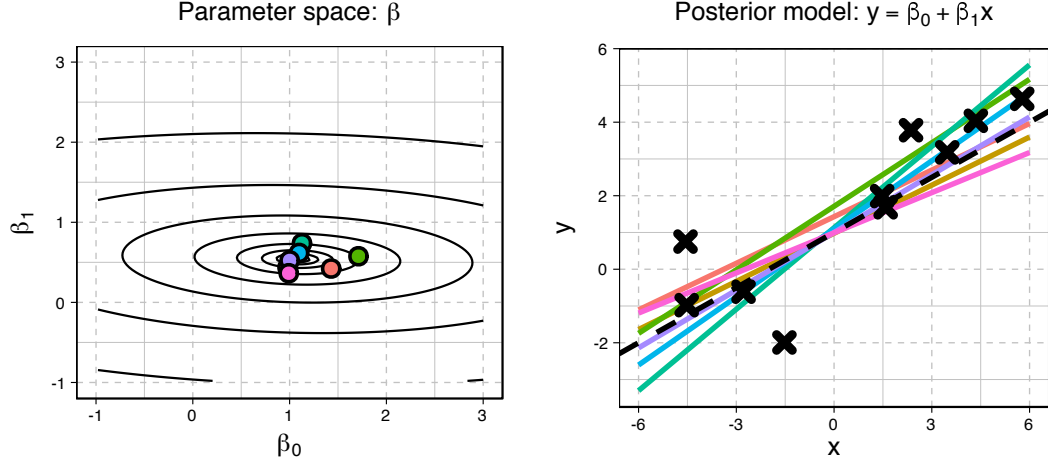
$$p(\beta|X, \mathbf{y}) = \mathcal{N}(\beta_1, \Sigma_1) \quad (2.9)$$

where  $\beta_1 = \sigma^{-2}\Sigma_1 X^\top \mathbf{y}$ , and  $\Sigma_1^{-1} = \sigma^{-2}X^\top X + \Sigma_0^{-1}$ . In order to make predictions for an unseen data point  $\mathbf{x}_*$ , the posterior predictive distribution for  $y_*$  is calculated as

$$p(y_*|\mathbf{x}_*, X, \mathbf{y}) = \int p(y_*|\mathbf{x}_*, \beta) p(\beta|X, \mathbf{y}) d\beta, \quad (2.10)$$

where, as before, the parameter  $\beta$  is marginalised. The difference is that the marginalisation is performed under the posterior distribution. This effectively takes into account all remaining uncertainty after assimilating the available data.

As before, every realisation of  $\beta$  defines a posterior linear model as shown in Figure 2.3. However, at this stage, the random nature of  $\beta$  is governed by the posterior. That is, after updating our prior beliefs in the light of data. The resulting distribution (2.10) is the predictive posterior distribution. The posterior form of  $\beta$  being a Gaussian and the Gaussian assumption on the response of the



**Figure 2.3:** *Posterior inference after observing data. The crosses shown in the right panel correspond to the data used for updating the prior. As before, the left panel shows the contour levels from the posterior distribution. Each sample drawn from the posterior induces a linear model on the right panel. Colours are used to identify each induced linear model with the corresponding sample. Note that the posterior reflects what is learned from the data. A linear model with positive slope and positive intercept.*

linear model, yield together the following analytic expression for the predictive posterior

$$p(y_* | \mathbf{x}_*, X, \mathbf{y}) = \mathcal{N}(\boldsymbol{\beta}_1^\top \mathbf{x}_*, \mathbf{x}_*^\top \Sigma_1 \mathbf{x}_*). \quad (2.11)$$

Moreover, this can be readily extended to make predictions in more than one point at a time. For example, assume there is interest in predicting the response for unseen points  $\mathbf{x}_*$  and  $\mathbf{x}_{**}$ . The joint predictive distribution can be written as

$$\begin{bmatrix} y_* \\ y_{**} \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} \boldsymbol{\beta}_1^\top \mathbf{x}_* \\ \boldsymbol{\beta}_1^\top \mathbf{x}_{**} \end{bmatrix}, \begin{bmatrix} \mathbf{x}_*^\top \Sigma_1 \mathbf{x}_* & \mathbf{x}_*^\top \Sigma_1 \mathbf{x}_{**} \\ \mathbf{x}_{**}^\top \Sigma_1 \mathbf{x}_* & \mathbf{x}_{**}^\top \Sigma_1 \mathbf{x}_{**} \end{bmatrix} \right). \quad (2.12)$$

Note that the predictive posterior induces a correlation based on the predictors  $\mathbf{x}_*$  and  $\mathbf{x}_{**}$ . This correlation can be written as  $k(\mathbf{x}_*, \mathbf{x}_{**}) = \mathbf{x}_*^\top \Sigma_1 \mathbf{x}_{**}$ . This observation is important in the discussion of the Gaussian process model.

The preceding discussion can be extended for a more flexible linear model. The linear expression in (2.5) can be written in terms of  $p$  known basis functions  $h_j(\mathbf{x})$ , with  $j = 1, \dots, p$ . The likelihood can be written as

$$p(\mathbf{y}|X, \boldsymbol{\beta}) = \mathcal{N}(H\boldsymbol{\beta}, \sigma^2 I_n), \quad (2.13)$$

where  $H$  is known as the design matrix with each row being equal to a vector of the form  $\mathbf{h}_i = \mathbf{h}(\mathbf{x}_i) = [h_1(\mathbf{x}_i), \dots, h_p(\mathbf{x}_i)]$ . The posterior distribution of  $\boldsymbol{\beta}$  is equal to

$$p(\boldsymbol{\beta}|X, \mathbf{y}) = \mathcal{N}(\boldsymbol{\beta}_1, \Sigma_1) \quad (2.14)$$

where  $\boldsymbol{\beta}_1 = \sigma^{-2} \Sigma_1 H^\top \mathbf{y}$ , and  $\Sigma_1^{-1} = \sigma^{-2} H^\top H + \Sigma_0^{-1}$ . Finally, the posterior predictive distribution for an unseen point  $\mathbf{x}_*$  can be shown to be

$$p(y_*|\mathbf{x}_*, X, \mathbf{y}) = \mathcal{N}(\boldsymbol{\beta}_1^\top \mathbf{h}_*, \mathbf{h}_*^\top \Sigma_1 \mathbf{h}_*), \quad (2.15)$$

where  $\mathbf{h}_* = \mathbf{h}(\mathbf{x}_*)$ . This allows the extension of the linear regression model with linear predictors to models with more flexible *known* basis functions.

In summary, a linear model with unknown coefficients  $\boldsymbol{\beta}$  has been specified to relate predictors  $\mathbf{x}$  and target variable  $y$ . The prior distribution on  $\boldsymbol{\beta}$  reflects our belief on which linear models are deemed possible. By combining likelihood, prior beliefs and data, the result is the posterior distribution on  $\boldsymbol{\beta}$ . This effectively reflects our updated beliefs in the linear model after observing some data.

### 2.2.2 The Gaussian process for regression

Formally, a Gaussian process (GP) is a stochastic process  $W = \{W_x : x \in \mathcal{X}\}$ , indexed by a set  $\mathcal{X}$ , such that the vector  $[W_{\mathbf{x}_1}, \dots, W_{\mathbf{x}_n}]^\top$  is distributed as a multivariate Gaussian for any choice of  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$  and  $n \in \mathbb{N}$ . In this sense, the Gaussian process is a generalisation of the multivariate Gaussian distribution. As its finite dimensional counterpart, it is completely determined by its mean and covariance functions. Let  $m(\mathbf{x})$  and  $k(\mathbf{x}, \mathbf{x}')$  denote the mean and covariance functions of a process  $f(\mathbf{x})$  such that

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]. \end{aligned} \quad (2.16)$$

The function  $f(\mathbf{x})$  is said to be distributed as a Gaussian process with mean  $m(\mathbf{x})$  and covariance  $k(\mathbf{x}, \mathbf{x}')$ , and it is denoted as



$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)). \quad (2.17)$$

Consider the linear model,  $f(\mathbf{x}) = \boldsymbol{\beta}^\top \mathbf{h}(\mathbf{x})$  with prior  $\boldsymbol{\beta} \sim \mathcal{N}(0, \Sigma)$ . It can be redefined as a Gaussian process with mean and covariance

$$\mathbb{E}[f(\mathbf{x})] = \mathbb{E}[\boldsymbol{\beta}]^\top \mathbf{h}(\mathbf{x}) = 0 \quad (2.18)$$

$$\mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = \mathbf{h}(\mathbf{x})^\top \mathbb{E}[\boldsymbol{\beta}\boldsymbol{\beta}^\top] \mathbf{h}(\mathbf{x}') = \mathbf{h}(\mathbf{x})^\top \Sigma \mathbf{h}(\mathbf{x}'). \quad (2.19)$$

The covariance function can be identified as  $k(x, x') = \mathbf{h}(x)^\top \Sigma \mathbf{h}(x')$ . Thus, the linear model can effectively be interpreted as a zero mean Gaussian process with covariance  $k(\mathbf{x}, \mathbf{x}') = \mathbf{h}(\mathbf{x})^\top \Sigma \mathbf{h}(\mathbf{x}')$ .

In general, the Gaussian process model is used as a prior over function spaces. That means that the function  $f(\cdot)$  is assumed to be a random realisation of a Gaussian process. In contrast with the linear model, the target variable  $y$  is considered as an output from an *underlying* function  $f(\cdot)$  with corresponding input variables  $\mathbf{x} \in \mathcal{X}$ . Assuming Gaussian errors for the observations, the complete Bayesian formulation is

$$y | f(\cdot), \mathbf{x} \sim \mathcal{N}(f(\mathbf{x}), \sigma^2) \quad (2.20)$$

$$f(\mathbf{x}) | \theta \sim \mathcal{GP}(m_\theta(\mathbf{x}), k_\theta(\mathbf{x}, \mathbf{x}')), \quad (2.21)$$

where  $\theta$  is a vector of parameters with some components used in the covariance function, and some in the mean function. For computational simplicity, the mean function is usually assumed to be 0, as in machine learning applications [186]. Alternatively, as is common in the computer code analysis literature, this can be seen as a modularised approach in which the Gaussian process is used to model the residuals. This partially mitigates the complexity of modelling the unknown function through a Gaussian process. This strategy has been found to be useful in identifying the discrepancy term in the analysis of computer models [142]. The covariance function, also known as covariance *kernel*, governs how the output from different locations are correlated as a function of the locations themselves. Common choices for the covariance function assume an isotropic model. That means, that the kernel is only a function of the Euclidean distance between points, and is not influenced by location or any given rotation. Popular choices are the Matérn, or the squared exponential. The latter assumes an infinite differentiable

process which in certain applications is considered a vague specification [204]. In contrast, the Matérn kernel contains a parameter that controls the differentiability of the implied functions. Other choices are possible and can accommodate for different prior information regarding the function to be estimated. For example, one dimensional spline kernels can be used to prevent the GP reverting back to a constant mean function when extrapolating data. If certain periodicity is suspected from the model, a periodic kernel might be employed [71].

Different isotropic kernels are shown in Figure 2.4. For a kernel to be isotropic, the kernel needs to satisfy the relation  $k(\mathbf{x}, \mathbf{y}) = k(\|\mathbf{x} - \mathbf{y}\|)$ . Thus, the kernel can be plotted against distance values as in the left panel of 2.4. The examples correspond to the Matérn kernel which is defined as

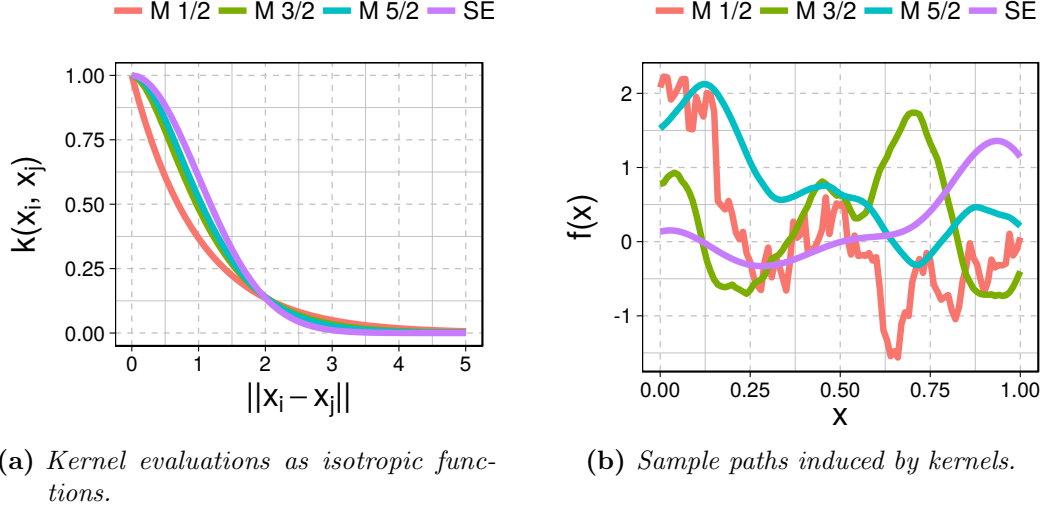
$$k(d) = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \sqrt{2\nu} \frac{d}{\varphi} \right)^\nu B_\nu \left( \sqrt{2\nu} \frac{d}{\varphi} \right), \quad (2.22)$$

where  $\Gamma(\cdot)$  is the gamma function,  $B_\nu(\cdot)$  is the modified Bessel function of the second kind,  $d = \|\mathbf{x} - \mathbf{y}\|$  is the distance between the kernel arguments, and  $\varphi$  and  $\nu$  are the lengthscale and smoothness parameter of the covariance function. Additionally, the squared exponential kernel is shown in Figure 2.4, which is defined as

$$k(d) = \sigma_f^2 \exp \left( -\frac{d^2}{2\varphi^2} \right), \quad (2.23)$$

where  $\varphi^2$  denotes the lengthscale parameter of the kernel. As shown in Figure 2.4, the squared exponential decays faster than the other kernels. Moreover, the Matérn kernels possess a rougher behaviour near the origin. That is, for points closer to each other. A sample function derived from each kernel is depicted in the right panel. As noted before, the Matérn kernel produces rougher sample paths. The Matérn kernel with smoothness parameter  $\nu = 1/2$ , also known as a exponential kernel, exhibits the most erratic behaviour. For more details on kernels and composition of kernels for Gaussian processes refer to Chapter 4 in [186]. Further discussion on additive and periodic kernels can be found in [70],[72],[71], and [97].

Once the kernel and all prior specification of the Gaussian process has been established the posterior can be computed. After observing some training input-output data, denoted by  $\mathcal{D}$ , the prediction is performed under the Bayesian



**Figure 2.4:** Different kernels are shown in the left panel as a function of the distance of its arguments. Note the lighter tails as the kernel becomes smoother. The Matérn 1/2 kernel possesses both heavier tails and rougher correlation near the origin. Sample paths induced by each kernel are shown on the right panel. Colours are used to identify each kernel. Note the smoother sample path from the squared exponential against the rougher sample paths from the Matérn variants.

framework. Let  $f_*$  denote the predicted output for unseen input  $\mathbf{x}_*$ , the joint distribution of the data and the unseen output is

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N} \left( 0, \begin{bmatrix} K + \sigma^2 I_n & k_* \\ k_*^\top & K_* \end{bmatrix} \right), \quad (2.24)$$

where  $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$  denotes the covariance matrix using the kernel and training data,  $k_* = [k(\mathbf{x}_1, \mathbf{x}_*), \dots, k(\mathbf{x}_n, \mathbf{x}_*)]^\top$  is the vector of cross-covariances between training data and  $\mathbf{x}_*$ , and  $K_* = k(\mathbf{x}_*, \mathbf{x}_*)$ . The joint distribution results in a Gaussian as a direct consequence of the finite dimensional properties of a Gaussian process. Thus, conditioning on the training data the predictive posterior distribution is

$$f_* | X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(\bar{f}_*, \bar{\sigma}^2(\mathbf{x}_*)), \quad (2.25)$$

where the mean predicted value is  $\bar{f}_* = k_*^\top [K + \sigma^2 I_n]^{-1} \mathbf{y}$ , and the variance is  $\bar{\sigma}^2(\mathbf{x}_*) = K_* - k_*^\top [K + \sigma^2 I_n]^{-1} k_*$ .

An example of a Gaussian process used in regression is depicted in Figure 2.5. The top panel shows samples from a zero mean Gaussian process prior with a square exponential kernel. The parameters are chosen as  $\sigma_f^2 = 1$  and  $\varphi = 1$ . The darker solid line shows the mean of the Gaussian process prior and the shaded regions correspond to credible bands for the expected function. The lower panel shows the Gaussian process posterior after observing some data, depicted as dots. The underlying true function is shown as the black dashed line. The observations were produced with an observation Gaussian model with variance  $\sigma^2 = 0.2$ . The dark blue solid line depicts the mean function from the Gaussian process posterior, and the shaded regions illustrate the corresponding credible regions.

For the Gaussian process posterior, note that the predicted mean can also be written as

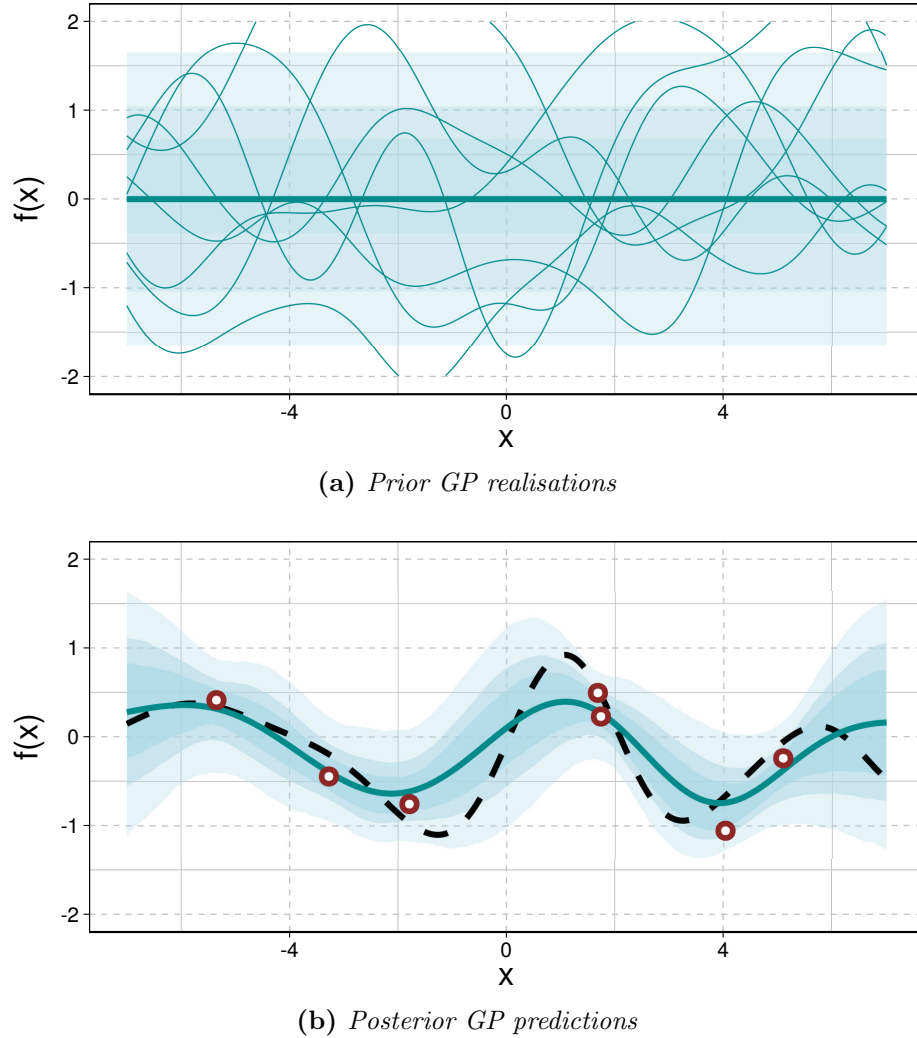
$$\begin{aligned}\bar{f}_* &= \mathbf{y}^\top [K + \sigma^2 I_n]^{-1} k_* \\ &= \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x}_*),\end{aligned}\tag{2.26}$$

with coefficients  $\alpha_i = [\mathbf{y}^\top (K + \sigma^2 I_n)^{-1}]_i$ . Being able to write the predicted value as above is evidence of the connection of Gaussian processes and *Reproducing Kernel Hilbert spaces*. The latter is an object of study in functional analysis. The Representer theorem in functional analysis guarantees equation (2.26) as a solution in function approximation theory using second order stochastic processes such as the Gaussian process [26, 1]. Additionally, the expression for the mean as in (2.26), together with Mercer's Representation theorem [see Section 3.2 26] can be used to determine an appropriate choice of kernel. The benefit is that a zero mean Gaussian process can be defined and all inferential power be treated through the covariance kernel. It is important to note that the kernel function by itself induces certain regularities and desirable properties in the mean of predictions.

To see this, Mercer's Representation theorem states that any kernel admits an expansion as a series of products of an orthonormal system of eigenfunctions and eigenvalues. That is,

$$k(\mathbf{x}, \mathbf{x}') = \sum_{k=1}^{\infty} \lambda_k \phi_k(\mathbf{x}) \phi_k(\mathbf{x}'),\tag{2.27}$$

where  $\phi_k(\cdot)$  is a sequence of orthonormal functions with associated eigenvalues  $\lambda_k$ .



**Figure 2.5:** A Gaussian process with squared exponential kernel is assumed for this example. Prior function realisations are shown with different curves in the top panel. The solid line shows the mean function from the GP prior. After observing some data, the GP prior is updated as shown in the lower panel. The solid line depicts the posterior mean from the GP model. The dashed line depicts the true underlying function with training samples shown as dots. The shaded regions show the interior error estimates from the GP posterior.

For example, consider the squared exponential kernel

$$k(x, x') = \exp\left(-\frac{1}{2\ell}(x - x')^2\right), \quad (2.28)$$

where  $x, x' \in \mathbb{R}$ , and  $\ell > 0$  denotes the lengthscale hyperparameter. This kernel induces an expansion for the predictions in terms of damped polynomials  $\psi_t(\mathbf{x}) = x^t \exp(-x^2/2\ell)$ . This type of polynomial expansion in the solution can be used for an infinite differentiable function with unbounded domains. Minka [155] provides an exposition on polynomial kernels and other approaches to solve for quadrature rules in integration problems with Gaussian processes as a prior over the function of interest. Moreover, explicit connections of mean functions and covariance kernels can also be consulted in [155].

In summary, the Gaussian process can be understood as a prior distribution on function space. Just as for the multivariate Gaussian distribution, it is completely determined by the mean and the variance-covariance functions. The covariance matrix is formed by evaluating the kernel function in every pair of points available for training. For computational convenience, it is often assumed a zero mean function for the process, and all predictive power is transferred to the choice of covariance kernel. This is because implicit assumptions about the mean of the process can be specified by selecting the kernel. For this reason, Stein [204] motivates the use of zero mean Gaussian processes with careful considerations on the kernel being used. In this dissertation, only the functional form of the kernel is chosen. The parameters of the kernel are subject to an inference procedure appropriate for Bayesian analysis, which is often a complicated task. Proper inference on the hyperparameters of the kernel is later discussed in Chapters 3 and 4, where sampling methods are presented to sample from such complicated posteriors.

## 2.3 Discussion

This chapter has presented an overview on the theoretical foundations of the methodology used in this dissertation. An overview in Bayesian inference for statistical models was discussed in Section 2.1. It was discussed that both the construction of models, and the mechanism for updating beliefs under the Bayesian framework provides a coherently systematic approach for uncertainty quantification

in engineering applications. Foundational texts on Bayesian inference can be found in [194] and [27]. Specialised texts in data analysis using Bayesian inference can be found in [138],[90], and [152].

Section 2.2 presents an overview on Gaussian processes as a Bayesian model for regression. The Gaussian process provides a highly flexible model that has proven to be useful even when small datasets are available. This property allows the Gaussian process to be a suitable emulator for computationally expensive computer codes. This has motivated the use of Gaussian processes as surrogates in this dissertation. Additionally, note that the monograph on Gaussian processes [186] has popularised the model in supervised learning applications as regression and classification. For a review of Gaussian processes used in spatial data analysis see [1, 204]. Berlinet and Thomas-Agnan [26] provide a study of properties of Hilbert spaces and their connections to second order stochastic processes such as the Gaussian process. Finally, Smola and Schölkopf [200] present the Gaussian process model in the context of Bayesian kernels methods for machine learning.

As discussed in Chapter 1, the hyperparameters of a Gaussian process are usually not known. However, due to computational convenience they are estimated via optimisation routines. In engineering applications where Gaussian processes are used as emulators the estimation of the hyperparameters provides only a myopic version of the Gaussian process posterior. This is because in engineering applications, where the function that is being modelled is computationally expensive, only a limited amount of data is available. This limitation coupled with a single candidate of hyperparameters does not allow one to infer the complete posterior distribution of the Gaussian process. In Chapters 3 and 4 two new sampling algorithms are presented to perform a complete Bayesian specification of the posterior distribution of a Gaussian process and its hyperparameters. This leads to robust uncertainty quantification in engineering applications where fast but accurate surrogates are needed. Moreover, in Chapter 5 the Gaussian process is used as a surrogate model in history matching. This represents a more restrictive setup in calibration of computer codes as expensive experimental replications only allow a handful of available data. In summary, the Bayesian framework for stating and updating probabilities allows for proper propagation of uncertainty as needed and exploited in subsequent chapters.





## CHAPTER 3

---

### Gaussian Process Hyperparameters: Parallel Asymptotically Independent Markov Sampling<sup>1</sup>

---

Computationally expensive computer codes are frequently needed to implement mathematical models which are assumed to be reliable approximations of physical processes in engineering. Such simulators often require intensive use of computational resources that makes them inefficient if further exploitation of the code is needed, *e.g.* optimisation, uncertainty propagation and sensitivity analysis [80, 130]. For this reason, surrogate models are needed to perform fast approximations to the output of demanding simulators and enable efficient exploration and exploitation of the input space. In this context, Gaussian processes are a common choice to build statistical surrogates -also known as *emulators*- which allow one to take into account the uncertainty derived from the inability to evaluate the original model in the whole input space. Gaussian processes have become popular in recent years due to their ability to fit complex mappings between outputs and inputs by means of a non-parametric hierarchical structure. Such applications are found, amongst many other areas, in computer code analysis [130], history matching [49], Machine Learning [186], Spatial Statistics [51] (with the name of Kriging), likelihood-free Bayesian Inference [221] and Genetics [124].

---

<sup>1</sup>The algorithms and ideas presented in this chapter have been published in Garbuno-Inigo et al. [85]

To build an emulator, a number of runs from the simulator is needed, but due to computing limitations only a small amount of evaluations can be performed. With a small amount of data, it is possible that the uncertainty of the parameters of the model cannot be described by a clearly uni-modal distribution. In such scenarios, Model Uncertainty Analysis [68] is capable of setting a proper framework in which we acknowledge all uncertainties related to the idealisations made through the modelling assumptions and the available, albeit limited information. To this end, *hierarchical modelling* should be taken into account. This corresponds to adding a layer of structural uncertainty to the assumed emulator either in a continuous or discrete manner [see 68, §4]. In the case of Gaussian processes, continuous structural uncertainty can be accounted for as a natural by-product from a Bayesian procedure. Hence, this is pursued in this work by focusing on samplers capable of exploring multi-modal distributions.

In order for the Gaussian process to be able to replicate the relation between inputs and outputs and make predictions, a training phase is necessary. Such training involves the estimation of the parameters of the Gaussian process from the data collected by running the simulator. These parameters are referred to as *hyperparameters*. The selection of the hyperparameters is usually done by using Maximum Likelihood estimates (MLE), or their Bayesian counterpart, Maximum a Posteriori estimates (MAP) [171, 186], or by sampling from the posterior distribution [222] in a fully Bayesian manner.

In this dissertation generating new runs from the simulator is assumed to be prohibitively expensive. Such limited information is not enough to completely identify either a candidate or a region of appropriate candidates for the hyperparameters. In this scenario, traditional optimisation routines [169] are not able to guarantee global optima when looking for the MLE or MAP, and a Bayesian treatment is the only option to account for all the uncertainties in the modelling. In the literature, however, it is common to see that MLE or MAP alternatives are preferred [130, 95] due to the numerical burden of maximising the likelihood function or because it is assumed that Bayesian integration will not produce results worth the effort. Though it is a strong argument in favour of estimating isolated candidates, in high-dimensional applications it is difficult to assess if the number of runs of the simulator is sufficient to produce robust hyperparameters. Robustness is usually measured with a prediction-oriented metric such as root-mean-square error (RMSE) [131], ignoring uncertainty and risk assessment of

---

choosing a single candidate of the hyperparameters by an inference process with limited data. In order to account for such uncertainty in the hyperparameters when making predictions, numerical integration should be performed. However, methods such as quadrature approximation become infeasible as the number of dimensions increases [130]. Therefore, an appropriate approach is to perform Monte Carlo integration [146]. This allows one to approximate any integral by means of a weighted sum, given a sample from the *correct* distribution.

In Gaussian processes, as in many other applications of statistics, the target distribution of the hyperparameters cannot be sampled directly and one should resort to Markov Chain Monte Carlo (MCMC) methods [188], as mentioned in Chapter 2. MCMC methods are powerful statistical tools but have a number of drawbacks if not tuned properly, particularly if one wishes to sample from multi-modal distributions [164, 110]. One of such limitations is the tuning of the proposal distribution, which allows the generation of new candidates for the chain. This proposal function has to be tuned with parameters that define its ability to move through the sample space. If an excessively wide spread is selected, this will produce samples with space-filling properties but which are likely to be rejected. On the other hand, having a narrower spread will cause an inefficient exploration of the sample space by taking short updates on the states of the chain, known in the literature as *Random Walk* behaviour [160]. In practice it is desirable to use a proposal distribution which is capable of balancing both extremes. Finding an appropriate tuning in high-dimensional spaces with sets of highly correlated variables can be an overwhelming task and often MCMC samplers can become expensive due to the long time needed to reach stationarity [45]. Neal [163] and Williams and Rasmussen [222] favour the Hybrid Monte Carlo (HMC) method to generate a sample from the posterior distribution. This prevents the random walk behaviour of traditional MCMC methods. If tuned correctly, the HMC should be able to explore most of the input space [144]. Such tuning process is problem-dependent and there is no guarantee that the method will sample from all existing modes, thus failing to adapt well to multi-modal distributions [166]. More details on HMC will be provided in Chapter 6 in the context of reliability analysis.

In this chapter, the Gaussian process model introduced in Chapter 2 is used in the context of computer code emulation. Also, a sampling algorithm is proposed for the marginalisation of the hyperparameters of a Gaussian process emulator.

The proposed algorithm uses the Transitional Markov Chain Monte Carlo (TMCMC) method of Ching and Chen [45] to set a framework for the parallelisation of Asymptotically Independent Markov Sampling in both the context of hyperparameter sampling (AIMS) [14] and in stochastic optimisation (AIMS-OPT) [226] reminiscent of Stochastic Subset Optimisation [209, 210]. Such an extension is built using concepts of Particle Filtering methods [6, 103], Adaptive Sequential Monte Carlo [56, 57] and Delayed Rejection Samplers [227, 156]. AIMS is chosen since it is an effective Sequential Monte Carlo sampler [161, 164, 56], which is a member of the family of Sequential Monte Carlo methods [67, 104]. It uses most of the information generated in the previous step in the sequence as opposed to traditional sequential methods, thus building a robust sampler when applied to multi-modal distributions. Finally, by using the AIMS-OPT algorithm a solution is built by means of a nested sequence of subsets, which converges to the optimal solution set. The algorithm can be terminated prematurely given a previously chosen accuracy threshold, thus providing a set of nearly optimal solutions. Whether it is composed by a single element, or a set of elements whose objective function differs by a negligible quantity, a full characterisation of the optimal solution is achieved. This contrasts with the capabilities of other stochastic optimisation schemes such as particle swarm optimisation or genetic algorithms [195].

By selecting the hyperparameters using the AIMS-OPT framework the effect is twofold. First, the uncertainty inherent to the specification of the hyperparameters is embedded in the set of suboptimal approximations to the solution. This uncertainty, expressed in a mixture of Gaussian process emulators, yields a robust surrogate where model uncertainty is accounted for. Second, computational implementation deficiencies of the inference procedure in Gaussian processes is overcome by incorporating stabilising approaches exposed in the literature as in Ranjan et al. [185], Andrianakis and Challenor [2] but in a Bayesian framework. The problem is therefore treated from both a probabilistic and an optimisation perspective.

The chapter is organised as follows. In Section 3.1, the Gaussian process (Chapter 2) is presented as an emulator for computationally expensive computer codes. Section 3.2 presents both the AIMS algorithm and the proper generalisation for a parallel implementation. Section 3.3 discusses several aspects of the computational implementation of the algorithm and their effect on the modelling assumptions. The result is the Parallel Asymptotically Independent Markov sampler (PAIMS).

The efficiency and robustness of PAIMS are discussed in Section 3.4 with some illustrative examples. A discussion of the results is given in Section 3.5.

### 3.1 The Gaussian Process as an Emulator

Let  $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  be the set of trials run by the simulator where  $\mathbf{x}_i \in \mathbb{R}^p$  denotes a given configuration for the model. The set  $X$  will be referred to as the set of *design points*. Let  $\mathbf{y} = \{y_1, \dots, y_n\}$  be the set of outputs observed for the design points. The pair  $(\mathbf{x}_i, y_i)$  will denote the *training run* being used to learn the emulator that approximates the simulator. The emulator is assumed to be a real-valued mapping  $\eta : \mathbb{R}^p \rightarrow \mathbb{R}$  which is an interpolator of the training runs, *i.e.*  $y_i = \eta(\mathbf{x}_i)$  for all  $i = 1, \dots, n$ . This omits any random error in the output of the computer code in the observed simulations, that is, the simulator is deterministic. It is assumed that the output of the simulator can be represented by a Gaussian process. Therefore, the set of design points is assumed to have a joint Gaussian distribution where the output satisfies the structure

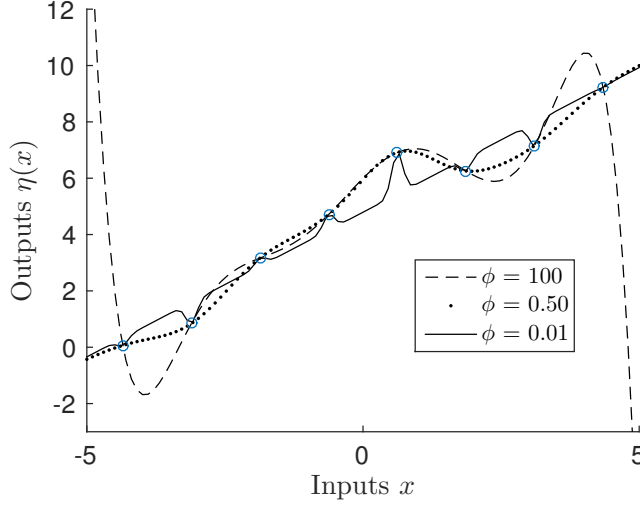
$$\eta(\mathbf{x}) = h(\mathbf{x})^T \boldsymbol{\beta} + Z(\mathbf{x} | \sigma^2, \boldsymbol{\phi}), \quad (3.1)$$

where  $h(\cdot)$  is a vector of known basis (location) functions of the input,  $\boldsymbol{\beta}$  is a vector of regression coefficients, and  $Z(\cdot | \sigma^2, \boldsymbol{\phi})$  is a Gaussian process with zero mean and covariance function

$$\text{cov}(Z(\mathbf{x}), Z(\mathbf{x}') | \sigma^2, \boldsymbol{\phi}) = \sigma^2 k(\mathbf{x}, \mathbf{x}' | \boldsymbol{\phi}), \quad (3.2)$$

where  $\sigma^2$  is the signal noise and  $\boldsymbol{\phi} \in \mathbb{R}_+^p$  denotes the *length-scale* parameters of the correlation function  $k(\cdot, \cdot)$ . Note that for a pair of design points  $(\mathbf{x}, \mathbf{x}')$ , the function  $k(\cdot, \cdot | \boldsymbol{\phi})$  measures the correlation between  $\eta(\mathbf{x})$  and  $\eta(\mathbf{x}')$  based on their respective input configurations. The effect of different values of  $\boldsymbol{\phi}$  in a one-dimensional example is depicted in Figure 3.1.

The role of the correlation function is to measure how close to each other the design points are, following the assumption that similar input configurations should produce similar outputs. For its analytical simplicity, interpretation and smoothness properties, this work uses the squared-exponential correlation function, namely



**Figure 3.1:** The length-scale parameters represent how sensitive is the output of the simulator to variations in each dimension. The plot corresponds to 8 design points chosen for the function  $\eta(x) = 5 + x + \cos(x) + .5 \sin(3x)$ . For low values of the length-scale parameter the training runs are less dependent of each other.

$$k(\mathbf{x}, \mathbf{x}' | \boldsymbol{\phi}) = \exp \left\{ -\frac{1}{2} \sum_{i=1}^p \frac{(x_i - x'_i)^2}{\phi_i} \right\}. \quad (3.3)$$

Note that other authors prefer the parametrisation with  $\phi_i^2$  as denominator. The reason behind this is that the squared parametrisation helps to interpret the length-scale parameter in the original scale of the data. However, this work uses a linear term in the denominator since the restriction of the length-scale parameters to lie in the positive orthant is more natural, as weights in the norm used to measure closeness and sensitivity to changes in such dimensions. Both interpretability and numerical performance can be improved if the length-scales refer to the same units, which leads to rescaling all dimensions of the input configurations. In the computer simulation terminology this translates in utilising experimental designs restricted to hypercubes, such as Latin hypercube sampling or Sobol sequences. Design of experiments is an active area of research outside the scope of this work.

In summary, the output of a design point, given the parameters  $\boldsymbol{\beta}, \sigma^2$  and  $\boldsymbol{\phi}$ , has a Gaussian distribution

$$y|\mathbf{x}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi} \sim \mathcal{N}(h(\mathbf{x})^T \boldsymbol{\beta}, \sigma^2 k(\mathbf{x}, \mathbf{x}'|\boldsymbol{\phi})), \quad (3.4)$$

which can be rewritten as the joint distribution of the vector of outputs  $\mathbf{y}$  conditional on the design points  $X$  and hyperparameters  $\boldsymbol{\beta}, \sigma^2$  and  $\boldsymbol{\phi}$  as

$$\mathbf{y}|X, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi} \sim \mathcal{N}(H\boldsymbol{\beta}, \sigma^2 K), \quad (3.5)$$

where  $H$  is the *design matrix* whose rows are the inputs  $h(\mathbf{x}_i)^T$  and  $K$  is the correlation matrix with elements  $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j|\boldsymbol{\phi})$  for all  $i, j = 1, \dots, n$ .

### 3.1.1 Estimating the Hyperparameters

The parameters of the process are not known beforehand and this induces uncertainty in the emulator itself. They can be estimated by Maximum Likelihood principles, but doing so lacks rigorous uncertainty quantification by concentrating all the density of the unknown quantities in a single value. The alternative is to treat them in a fully Bayesian manner and marginalise them when performing predictions. This way their respective uncertainty is taken into account. In this scenario, the prediction  $y^*$  for a non-observed configuration  $\mathbf{x}^*$  can be performed with the data available,  $\mathcal{D} = (\mathbf{y}, X)$ , and the evidence they shed on the parameters of the Gaussian process. Therefore, the predictions should be made with the marginalised posterior distribution

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \int_{\Theta} p(y^*|\mathbf{x}^*, \mathcal{D}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta}, \quad (3.6)$$

where  $\boldsymbol{\theta} = (\boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi})$  denotes the complete vector of hyperparameters. One should note that given the properties of a collection of Gaussian random variables, a prediction for  $y^*$  conditioned in the data and  $\boldsymbol{\theta}$  is also a Gaussian random variable [see 171]. As in hierarchical modelling, each possible value of  $\boldsymbol{\theta}$  defines a specific realisation of a Gaussian distribution, so it is appropriate to refer to  $\boldsymbol{\theta}$  as the hyperparameters of the Gaussian process.

Due to its computational complexity, the integral in (3.6) is often omitted when making predictions. It is commonly assumed that the MLE of the likelihood

$$\mathcal{L}(\boldsymbol{\theta}) = p(\mathbf{y}|X, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi}), \quad (3.7)$$

or the MAP estimate from the posterior distribution

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathbf{y}|X, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi}) p(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi}), \quad (3.8)$$

are robust enough to account for all the uncertainty in the modelling. However, when either the likelihood (3.7) is a non-convex function or the posterior (3.8) is a multi-modal distribution, conventional optimisation routines might only find local optima, thus failing to find the most probable candidate of such distribution. Moreover, by selecting only one candidate, robustness and uncertainty quantification are lost in the process. Additionally, there are degenerate cases when it is crucial to estimate the integral in (3.6) by means of Monte Carlo simulation instead of by proposing a single candidate. As it has been noted by Andrianakis and Challenor [2], two extreme cases for the Gaussian process length-scale hyperparameters can be identified. One possibility is for  $\phi$  to approach infinity, which makes every design point dependent on each other; the other, when  $\phi$  approaches the origin where a multivariate regression model becomes the limiting case. In the first case, high correlation among all the training runs results in a model which is not able to distinguish local dependencies. As for the second, it violates the assumptions that constitute a Gaussian process, by completely ignoring the correlation structure in the design points to predict the output. Consequently, if MCMC is performed one can approximate the integrated predictive distribution in (3.6) by means of

$$p(y^*|\mathbf{x}^*, \mathcal{D}) \approx \sum_{i=1}^N w_i p(y^*|\mathbf{x}^*, \mathcal{D}, \boldsymbol{\theta}_i), \quad (3.9)$$

where  $\boldsymbol{\theta}_i$  is obtained through an appropriate sampler, *i.e.* one capable of sampling from multi-modal distributions. The coefficients  $w_i$  denote the weights of each sample generated. Since each term  $p(y^*|\mathbf{x}^*, \mathcal{D}, \boldsymbol{\theta}_i)$  corresponds to a Gaussian density function, the predictions are made by a mixture of Gaussians.

**Theorem 1.** *If the emulator output  $y^*$  conditional on its configuration vector  $\mathbf{x}^*$  has a posterior density as in (3.9), then its mean function and covariance function can be computed as*



$$\mu(\mathbf{x}^*) = \sum_{i=1}^N w_i \mu_i(\mathbf{x}^*), \quad (3.10)$$

$$\text{cov}(\mathbf{x}^*, \mathbf{x}') = \sum_{i=1}^N w_i [(\mu_i(\mathbf{x}^*) - \mu(\mathbf{x}^*))(\mu_i(\mathbf{x}') - \mu(\mathbf{x}')) + \text{cov}(\mathbf{x}^*, \mathbf{x}' | \boldsymbol{\theta}_i)], \quad (3.11)$$

where  $\mu_i(\mathbf{x}^*)$  denotes the expected value of the likelihood distribution of  $y^*$  conditional on the hyperparameters  $\boldsymbol{\theta}_i$ , the training runs  $\mathcal{D}$  and the input configuration  $\mathbf{x}^*$ .

*Proof.* Equality in (3.10) is a direct application of the tower property of conditional expectation and (3.11) follows from the covariance decomposition formula using the vector of weights  $w_i$  as an auxiliary probability distribution on the conditioning. ■

From equation (3.11) we can compute the variance, also known as the prediction error, of an untested configuration  $\mathbf{x}^*$  as

$$\sigma^2(\mathbf{x}^*) = \sum_{i=1}^N w_i ((\mu_i(\mathbf{x}^*) - \mu(\mathbf{x}^*))^2 + \sigma_i^2(\mathbf{x}^*)). \quad (3.12)$$

By doing this, a more robust estimation of the prediction error is made since it balances the predicted error in one sample with how far the prediction of such sample is from the overall estimation of the mixture.

### 3.1.2 Prior Distributions

In order to perform a Bayesian treatment for the prediction task in equation (3.6) the prior distribution  $p(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi})$  in equation (3.8) has to be specified. Weak prior distributions are commonly used for  $\boldsymbol{\beta}$  and  $\sigma^2$  [171]. Such weak prior has the form

$$p(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\phi}) \propto \frac{p(\boldsymbol{\phi})}{\sigma^2}, \quad (3.13)$$

where it is assumed a priori that both the covariance and the mean hyperparameters are independent. Even more,  $\boldsymbol{\beta}$  and  $\sigma^2$  are assumed to have an improper non-informative distribution.

As for the length-scale hyperparameter  $\boldsymbol{\phi}$ , a prior distribution  $p(\boldsymbol{\phi})$  is still needed. In this case the reference prior [studied by 22, 24] sets an objective framework to account for the uncertainty of  $\boldsymbol{\phi}$ , thus avoiding any potential bias induced

by the modelling assumptions. This prior is built based on Shannon’s expected information criteria and allows the use of a prior distribution in a setting where no previous knowledge is assumed. That way, the training runs are the only source of information for the inference process. Additionally, the reference prior is capable of ruling out subspaces of the sample space of the hyperparameters [5], thus reducing regions of possible candidates of Gaussian distributions in the mixture model in equation (3.9). Since this provides an off-the-shelf framework for the estimation of the hyperparameters, the reference prior developed by Paulo [179] is used in this work. However, there are no known analytical expressions for its derivatives which limits its application to MCMC samplers that use gradient information. Note that there are other possibilities available for the prior distribution of  $\phi$ . Examples of these are the log-normal or log-Laplacian distributions, which can be interpreted as a regularisation in the norm of the parameters. Andrianakis and Challenor [5] suggest a decaying prior. Another option is to elicit prior distributions from expert knowledge as in Oakley [172].

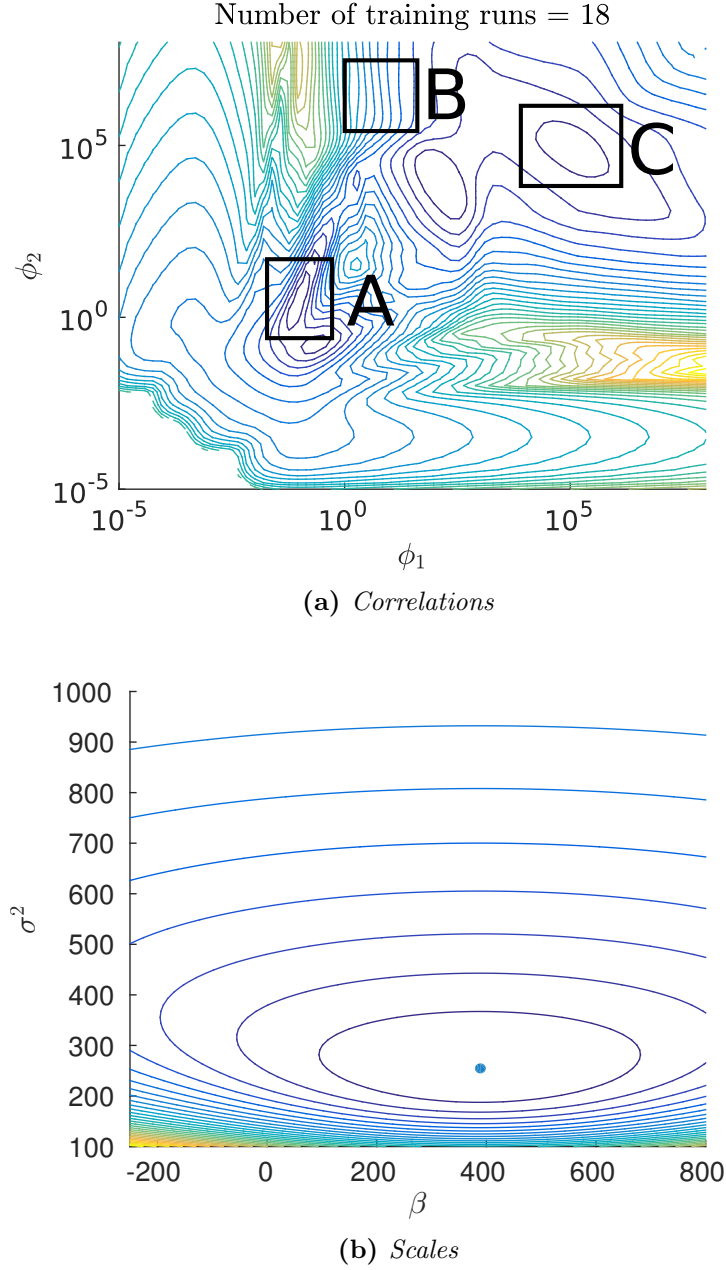
### 3.1.3 Marginalising the Nuisance Hyperparameters

The nature of the hyperparameters  $\beta, \sigma^2$  and  $\phi$  is potentially different in terms of scales and dynamics, as seen and explained in Figure 3.2. It is possible to cope with these limitations by using a Gibbs sampling framework, but it is well-known that such sampling scheme can be inefficient if it is used for multi-modal distributions in higher dimensions. Analogously, a Metropolis-Hastings sampler can also be overwhelmed.

Another alternative is to focus on  $\phi$  and perform the inference in the correlation function. This is done by regarding  $\beta$  and  $\sigma^2$  as nuisance parameters and integrating them out from the posterior distribution (3.8). The modelling assumptions in the training runs and the prior distribution, equations (3.5) and (3.13) respectively, allow one to identify a Gaussian-inverse-gamma distribution for  $\beta$  and  $\sigma^2$ , which can be shown to yield the integrated posterior distribution

$$p(\phi|\mathcal{D}) \propto p(\phi) (\hat{\sigma}^2)^{-\frac{n-p}{2}} |K|^{-\frac{1}{2}} |H^T K^{-1} H|^{-\frac{1}{2}}, \quad (3.14)$$

where



**Figure 3.2:** In 3.2a, different dynamics of the hyperparameters for the log-posterior distribution of test function 3.4.1 are shown: **A.** corresponds to positive correlation between  $\phi_1$  and  $\phi_2$ . **B.** corresponds to an independent region. **C.** corresponds to negative correlation between  $\phi_1$  and  $\phi_2$ . In 3.2b, the marginal log-posterior function of the same example with  $h(x) = 1$ , presents the same contour level for a wide range of  $\beta$ . Thus, the hyperparameters exhibit very different scales. The dot represents the minimum of the corresponding function.

$$\hat{\sigma}^2 = \frac{\mathbf{y}^T (K^{-1} - K^{-1}H(H^T K^{-1}H)^{-1}H^T K^{-1}) \mathbf{y}}{n - p - 2}, \quad (3.15)$$

and

$$\hat{\boldsymbol{\beta}} = (H^T K^{-1}H)^{-1}H^T K^{-1}\mathbf{y}, \quad (3.16)$$

are estimators of the signal noise  $\sigma^2$  and regression coefficients  $\boldsymbol{\beta}$  [see 171, for further details]. Additionally, the predictive distribution conditioned on the hyperparameters follows a Gaussian distribution with mean and correlation functions

$$\mu(\mathbf{x}^*|\boldsymbol{\phi}) = h(\mathbf{x}^*)^T \hat{\boldsymbol{\beta}} + t(\mathbf{x}^*)^T K^{-1}(\mathbf{y} - H\hat{\boldsymbol{\beta}}), \quad (3.17)$$

$$\begin{aligned} \text{corr}(\mathbf{x}^*, \mathbf{w}^*|\boldsymbol{\phi}) &= k(\mathbf{x}^*, \mathbf{w}^*|\boldsymbol{\phi}) - t(\mathbf{x}^*)^T K^{-1} t(\mathbf{w}^*) + \\ &\quad (h(\mathbf{x}^*)^T - t(\mathbf{x}^*)^T K^{-1}H) (H^T K^{-1}H)^{-1} (h(\mathbf{w}^*)^T - t(\mathbf{w}^*)^T K^{-1}H)^T, \end{aligned} \quad (3.18)$$

where  $\mathbf{x}^*$ ,  $\mathbf{w}^*$  denote a pair of test configurations and  $t(\mathbf{x}^*)$  denotes the vector obtained by computing the covariance of the new proposal with every design point  $t(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1|\boldsymbol{\phi}), \dots, k(\mathbf{x}, \mathbf{x}_n|\boldsymbol{\phi}))^T$ . Note that both estimators depend only on the correlation function hyperparameters  $\boldsymbol{\phi}$  since both  $\boldsymbol{\beta}$  and  $\sigma^2$  have been integrated out. Considerations of when it is appropriate to integrate out the hyperparameters in a model has been discussed by MacKay [147]. In the Gaussian process context it gains additional significance since it allows the development of appropriate MCMC samplers capable of overcoming the dynamics of different sets of hyperparameters.

In the light of the above discussion, this work focuses on the inference drawn from the correlation function  $k(\cdot, \cdot)$  in equation (3.2), since the structure of dependencies of the training runs to predict the outputs is recovered by it. The main assumption is that the mean function hyperparameter  $\boldsymbol{\beta}$  contains minor information on the structural dependencies of the data, relative to the correlation function hyperparameters, which would prevent the use of integrated likelihoods [see 23, for further discussion]. If prior information is available, then an additional effort can be made on eliciting an appropriate mean function for the Gaussian process emulator. Such information can be related to expert knowledge of the simulator which eventually allows the analyst to build a better mean function by adding significant regression covariates [see 216, for a detailed discussion].

## 3.2 AIMS Framework

Hyperparameter marginalisation by means of Monte Carlo methods in Gaussian processes is usually performed by Hybrid Monte Carlo methods [163, 222] which are capable of suppressing the Random Walk behaviour of MCMC samplers if tuned correctly. In this work, the sampling of the hyperparameters is done by means of Asymptotically Independent Markov Sampling (AIMS) [14]. This method combines techniques developed for Bayesian inference such as Importance Sampling and Simulated Annealing [132] to sample from the posterior distribution as done by other MCMC algorithms. Additionally, AIMS can also be adapted for global optimisation (AIMS-OPT) [226] in a fashion of the traditional simulated annealing method for stochastic optimisation. Let the problem be

$$\min_{\phi \in \Phi} \mathcal{H}(\phi|\mathcal{D}), \quad (3.19)$$

where  $\mathcal{H}(\phi|\mathcal{D})$  denotes the negative log-posterior distribution conditional on the set of training runs  $\mathcal{D}$ . Let the set of optimal solutions to the optimisation problem above be

$$\Phi^* = \left\{ \phi \in \Phi : \phi = \arg \min_{\phi \in \Phi} \mathcal{H}(\phi|\mathcal{D}) \right\}, \quad (3.20)$$

where  $|\Phi^*| \geq 1$ . This formulation acknowledges the presence of multiple global optima in the posterior distribution conditional on the training runs. It is important to note that using the logarithm of the posterior distribution reduces the overflow in the computation of the equation (3.14), which is likely to arise due to ill-conditioning of the matrix  $K$  [165]. Ill-conditioned matrices are common in Gaussian process applications when using a squared exponential kernel. This is due to the assumptions that this type of kernel induces in the differentiability of the process and, subsequently, on the simulator being modelled. The squared exponential kernel induces greater correlations for the global behaviour of the model. The consequence is that training points can still have an effect on points which are farther apart. As it will be discussed in Section 3.3, a nugget term is added to the formulation to improve the stability of computations using, for example, a squared exponential covariance kernel, and to match a Bayesian formulation with an assumed observational error.

In this context, AIMS-OPT is capable of producing samples by means of a

sequence of nested subsets  $\Phi_{k+1} \subseteq \Phi_k$  that converges to the set of optimal solutions  $\Phi^*$ . Thus, if the algorithm is terminated in a premature step, a set of sub-optimal approximations to (3.20) will be recovered. Let  $\{p_k(\phi|\mathcal{D})\}_{k=1}^\infty$  be the sequence of density distributions such that

$$p_k(\phi|\mathcal{D}) \propto p(\phi|\mathcal{D})^{1/\tau_k} = \exp\{-\mathcal{H}(\phi|\mathcal{D})/\tau_k\}, \quad (3.21)$$

for a sequence of monotonically decreasing temperatures  $\tau_k$ . By tempering the distributions in this manner, the samples obtained in the first step of the algorithm are approximately distributed as a uniform random variable over a *practical support*, while in the last annealing level, they are distributed uniformly on the set of optimal solutions, namely

$$\lim_{\tau \rightarrow \infty} p_\tau(\phi|\mathcal{D}) = U_\Phi(\phi), \quad (3.22)$$

$$\lim_{\tau \rightarrow 0} p_\tau(\phi|\mathcal{D}) = U_{\Phi^*}(\phi), \quad (3.23)$$

where  $U_A(\phi)$  denotes a uniform distribution over the set  $A$  for every  $\phi \in A$ .

### 3.2.1 Annealing at Level $k$

The general framework for the AIMS-OPT algorithm is presented, focusing on how to sample from the hyperparameter space at level  $k$  based on the sample of the previous level. Let  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)}$  be samples of the hyperparameters distributed as  $p_{k-1}(\phi)$  at level  $k-1$ . For notational simplicity, the conditional on  $\mathcal{D}$  will be omitted from  $p_{k-1}(\cdot)$ , however the training runs are crucial to build statistical surrogates. The objective is to use a kernel such that  $p_k(\cdot)$  is the stationary distribution of the Markov chain. Let  $\mathcal{P}_k$  denote such Markov transition kernel, which satisfies the continuous Chapman-Kolmogorov equation

$$p_k(\phi) d\phi = \int_{\Phi} \mathcal{P}_k(d\phi|\xi) p_k(\xi) d\xi, \quad (3.24)$$

where  $p_k(d\phi) = p_k(\phi) d\phi$  denotes the probability measure. By applying importance sampling using the distribution at the previous annealing level, equation (3.24) can be approximated as

$$\begin{aligned}
p_k(\phi) d\phi &= \int_{\Phi} \mathcal{P}_k(d\phi|\xi) \frac{p_k(\xi)}{p_{k-1}(\xi)} p_{k-1}(\xi) d\xi \\
&\approx \sum_{j=1}^N \mathcal{P}_k(d\phi|\phi_j^{(k-1)}) \bar{\omega}_j^{(k-1)} = \hat{p}_{k,N}(d\phi),
\end{aligned} \tag{3.25}$$

where  $\hat{p}_{k,N}(\cdot)$  is used as the *global* proposal distribution for a candidate in the chain and

$$\omega_j^{(k-1)} = \frac{p_k(\phi_j^{(k-1)})}{p_{k-1}(\phi_j^{(k-1)})} \propto \exp \left\{ -\mathcal{H}(\phi_j^{(k-1)}|\mathcal{D}) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\}, \tag{3.26}$$

$$\bar{\omega}_j^{(k-1)} = \frac{\omega_j^{(k-1)}}{\sum_{j=1}^N \omega_j^{(k-1)}}, \tag{3.27}$$

are the importance weights and the normalised importance weights respectively. Note that for computing  $\bar{\omega}_j^{(k-1)}$  the normalising constant of the integrated posterior distribution (3.14) is not needed.

The proposals of candidates for the chain are done in two steps. In the first step, a candidate is drawn as an update from a random *marker* from the sample of the previous annealing level, checking whether it is accepted or not. If the local candidate is rejected by a Random Walk Metropolis-Hastings evaluation, then the chain remains invariant,  $\phi_{i+1}^{(k)} = \phi_i^{(k)}$ , and another marker is selected at random. In the second step, given the candidate has been accepted as a local proposal, such candidate is considered as being drawn from the approximation in (3.25) and accepted in an Independent Metropolis-Hastings framework, hence called a global candidate for the chain. Let  $q_k(\cdot|\cdot)$  denote the symmetric transition distribution used for local proposals for the Markov chain. The subscript  $k$  accounts for the adaptive nature of the transition steps in each annealing level. Thus, the kernel distribution of the Random Walk, which leaves the intermediate density invariant, can be written as

$$\mathcal{P}_k(d\phi|\xi) = q_k(\phi|\xi) \min \left\{ 1, \frac{p_k(\phi)}{p_k(\xi)} \right\} d\phi + (1 - \alpha_k(\xi)) \delta_{\xi}(d\phi), \tag{3.28}$$

where  $\delta_{\xi}(d\phi)$  denotes a delta density and  $\alpha_k(\xi)$  is the probability of accepting the transition from  $\xi$  to  $\Phi \setminus \{\xi\}$ . It follows from (3.25) that the approximated stationary condition of the target distribution at annealing level  $k$  can be written

as

$$\hat{p}_{k,N}(\phi) = \sum_{j=1}^N \bar{\omega}_j^{(k-1)} q_k \left( \phi \mid \phi_j^{(k-1)} \right) \alpha_k^l \left( \phi \mid \phi_j^{(k-1)} \right), \quad (3.29)$$

with

$$\alpha_k^l(\xi \mid \phi) = \min \left\{ 1, \frac{p_k(\xi)}{p_k(\phi)} \right\}, \quad (3.30)$$

the probability of accepting the local transition; whereas

$$\alpha_k^g(\xi \mid \phi) = \min \left\{ 1, \frac{p_k(\xi) \hat{p}_{k,N}(\phi)}{p_k(\phi) \hat{p}_{k,N}(\xi)} \right\}, \quad (3.31)$$

denotes the probability of accepting such candidate for the Markov chain, hence accepting a global transition [see 226, for a detailed discussion]. This leads to Algorithm 1 and 2 detailed below.

According to Algorithm 1, the initialising step should also be provided for the annealing level. In practical implementations it is suggested that it should be considered to be  $\phi_1^{(k)} \sim q_k(\phi \mid \phi_j^{(k-1)})$  where  $j = \arg \max_i \bar{\omega}_i^{(k-1)}$ , *i.e.* the sample with the largest normalised importance weight.

### 3.2.2 Adaptive Proposal Distribution and Temperature Scheduling

Even though a Random Walk is performed in every local proposal, AIMS-OPT performs efficient sweeping of the sample space by producing candidates from neighbourhoods of the markers from the previous annealing level  $\{\phi_j^{(k-1)}\}_{j=1}^N$ . This is accomplished if the transition distribution  $q_k(\phi \mid \phi_j^{(k-1)})$  uses an appropriate proposal distribution where sampling is to be realised; namely, the level curves of the tempered distribution. To be able to cope with the non-negative restriction and to neglect the effect of the scales on each dimension, the transitions are performed in the log-space of the length-scale parameters  $\phi$ , as suggested by Neal [162]. The symmetric transition distribution proposed is a Gaussian distribution for such log-parameters. That is, each local candidate will be distributed as

$$\xi \sim \mathcal{N} \left( \xi \mid \phi_j^{(k-1)}, c_k \Sigma_k \right), \quad (3.37)$$



**Algorithm 1:** AIMS-OPT at annealing level  $k$ **Input :**

- ◇  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)} \sim p_{k-1}(\phi)$ , generated at previous level;
- ◇  $\phi_1^{(k)} \in \Phi \setminus \{\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)}\}$ , initial state of the chain;
- ◇  $q_k(\phi|\xi)$ , symmetric local proposal;

**Output :**

- ◇  $\phi_1^{(k)}, \dots, \phi_N^{(k)} \sim p_k(\phi)$ ;

**for**  $i \leftarrow 2$  **to**  $n - 1$  **do**

- (1) Generate a local candidate using the previous level samples as “markers”

$$\begin{aligned} \xi &\sim Q_k \left( \xi \mid \phi_1^{(k-1)}, \dots, \phi_n^{(k-1)} \right) \\ &= \sum_{j=1}^N \bar{\omega}_j^{(k-1)} q_k \left( \xi \mid \phi_j^{(k-1)} \right) \end{aligned} \quad (3.32)$$

- a) Select index  $j$  with probability proportional to importance weights  $\omega_1^{(k-1)}, \dots, \omega_N^{(k-1)}$ .

- b) Generate candidate from the local proposal distribution

$$\xi \sim q_k \left( \xi \mid \phi_j^{(k-1)} \right) \quad (3.33)$$

- c) Accept  $\xi$  as a local candidate with probability

$$\alpha_k^l \left( \xi \mid \phi_j^{(k-1)} \right) \quad (3.34)$$

- (2) Update  $\phi_i^{(k)} \rightarrow \phi_{i+1}^{(k)}$  by accepting or rejecting  $\xi$  using Algorithm 2.

**end**

where  $c_k$  is a decaying parameter for the spread of the proposal, *i.e.*  $c_k = \nu c_{k-1}$  with  $\nu \in (0, 1)$  commonly chosen as  $\nu = 1/2$  [226]. The matrix  $\Sigma_k$  denotes the covariance matrix for log-parameters where typical choices can be the identity matrix  $I_{p \times p}$ , a diagonal matrix or a symmetric positive definite matrix. We propose the use of the weighted covariance matrix estimated from the sample and their importance weights of the previous level  $(\bar{\omega}_1^{(k-1)}, \phi_1^{(k-1)}), \dots, (\bar{\omega}_N^{(k-1)}, \phi_N^{(k-1)})$ . By doing so, the scale and directions of the ellipsoids of the Gaussian steps are learned

---

**Algorithm 2:** Global acceptance of  $\xi$ 


---

**if**  $\xi$  was accepted as local candidate **then**

    Accept  $\xi$  as a global transition with probability

$$\alpha_k^g \left( \xi \middle| \phi_i^{(k)} \right) \quad (3.35)$$

**else**

    Leave the chain invariant

$$\phi_{i+1}^{(k)} = \phi_i^{(k)} \quad (3.36)$$

**end**

---

as in Adaptive Sequential Monte Carlo methods [108, 77] from the information gathered from the previous level in the sequence.

The annealing sequence and its effective exploration of the sample space is dictated by the temperature  $\tau_k$  of the intermediate distributions. Moreover, it defines how different is one target distribution from the next one, so the effectiveness of the sample as markers from the previous annealing level depends strongly on how the scheduling is performed. It is clear that abrupt changes lead to rapid deterioration of the sample, whilst low paced changes could produce unnecessary steps in the annealing schedule. In order to cope with this compromise, Zuev and Beck [226] used the *effective sampling size* to determine the value of the next temperature in the process. That is solving for  $\tau_k$ , when a sample from level  $k - 1$  has been produced, in

$$\frac{\sum_{j=1}^n \exp \left\{ -2\mathcal{H}(\phi_j^{(k-1)}) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\}}{\left( \sum_{j=1}^n \exp \left\{ -\mathcal{H}(\phi_j^{(k-1)}) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\} \right)^2} = \frac{1}{\gamma n}, \quad (3.38)$$

where  $\gamma$  defines a threshold for the proportion of the sample to be as effective from the importance sampling. Note that the value of  $\gamma$  defines additionally how many annealing steps will be performed. As suggested from Zuev and Beck [226] a value of 1/2 is used for such parameter.

### 3.2.3 Stopping Condition

If the temperature continues to drop along the sequence of intermediate distributions, eventually an *absolute zero*  $\tau_k = 0$  would be reached. However, such limit cannot be achieved in practical implementations and a stopping condition

is needed for the algorithm. By the same assumptions as in the original paper [226] and without loss of generality, the objective function  $\mathcal{H}(\phi)$  is assumed to be non-negative. Similarly, let  $\delta_k$  denote the coefficient of variation (cov) of the sample  $\mathcal{H}(\phi_1^{(k)}), \dots, \mathcal{H}(\phi_N^{(k)})$ , *i.e.*

$$\delta_k = \frac{\sqrt{\frac{1}{N} \sum_{i=1}^N \left( \mathcal{H}(\phi_i^{(k)}) - \frac{1}{N} \sum_{j=1}^N \mathcal{H}(\phi_j^{(k)}) \right)^2}}{\frac{1}{N} \sum_{j=1}^N \mathcal{H}(\phi_j^{(k)})}. \quad (3.39)$$

Therefore,  $\delta_k$  is used as a measure of the sensitivity of the objective function to the hyperparameters in the domain  $\Phi_{\tau_k}^*$ . If the samples are all located in  $\Phi^*$  then their COV will be zero, since  $\forall j \mathcal{H}(\phi_j^{(k)}) = \min_{\phi \in \Phi^*} \mathcal{H}(\phi)$ . As the progression of the intermediate distributions advances with  $k$ , it is expected that  $\delta_k \rightarrow 0$ . As a consequence, a criteria to stop the annealing sequence is needed, and the algorithm will stop when the following condition is attained

$$\delta_k < \alpha \delta_0 = \delta_{\text{target}}, \quad (3.40)$$

where  $\alpha$  is assumed to be 0.10 in practical implementations to prevent the algorithm generating redundant annealing levels in the last steps of the procedure. Note that the stopping criterion (3.40) is used to drive the simulated annealing temperature towards the absolute zero. However, if the aim is not localising modes as in stochastic optimisation, and a more traditional oriented sampling is required, the algorithm could be truncated in a temperature value of 1. This adds an additional layer of flexibility to the algorithm which other stochastic-search approaches do not share.

### 3.2.4 Parallel Implementation and Guarding Against Rejection

As found in our earliest experiments, AIMS-OPT with the global acceptance rule as in Algorithm 2 might degenerate quickly in higher dimensions since the starting of the chain comes from the highest normalised weighted sample and a transition might take too long to be performed, resulting in high rejection rates. Furthermore, information from the markers is lost since they do not provide good transition neighbourhoods and the ability to create new samples for the next annealing level is maimed. This aside, AIMS-OPT can become computationally expensive when

the number of samples increases. To cope with these limitations it is proposed to incorporate the Transitional Markov Chain Monte Carlo (TMCMC) and the Delayed Rejection methods into the AIMS-OPT framework. This extension not only enhances the mixing properties of the sampler, *i.e.* improve acceptance rates, but also provides a computational framework in which parallel Markov chains can be sampled from the intermediate distributions  $p_k(\phi)$  of the length-scale hyperparameters.

The idea to enable parallelisation comes from the TMCMC algorithm [see 45, for further details]. In the framework of Algorithm 1, every marker from the annealing level  $k - 1$  is a starting point for a Markov chain. This produces not only specialised chains which are likely to explore the marker's neighbourhood on the sample space, but also allows an assessment of which markers will generate a better chain. The normalised weights  $\bar{\omega}_j^{(k)}$  will dictate how deeply a chain will evolve starting from its marker  $\phi_j^{(k-1)}$ . Consequently, the number of samples in each chain will be set with probability proportional to the normalised weight, a direct result from the TMCMC algorithm.

In order to guard against high rejection rates, and therefore degeneracy on the sampling scheme, it is proposed to generate an additional candidate if the first one is rejected as in Delayed Rejection Algorithms [156]. Let  $S_1(\cdot|\cdot)$ ,  $S_2(\cdot|\cdot, \cdot)$  be a one step and two steps proposal density distributions respectively;  $\pi(\cdot)$  the target distribution of the Markov chain and  $a_1(\cdot, \cdot)$  the probability of accepting a transition in one step. Then, the probability of accepting a transition in two steps, denoted by  $a_2(\cdot, \cdot)$ , is

$$a_2(\phi_0, \phi_2) = \min \left\{ 1, \frac{\pi(\phi_2) S_1(\phi_1|\phi_2) S_2(\phi_0|\phi_2, \phi_1) (1 - a_1(\phi_2, \phi_1))}{\pi(\phi_0) S_1(\phi_1|\phi_0) S_2(\phi_2|\phi_0, \phi_1) (1 - a_1(\phi_0, \phi_1))} \right\}, \quad (3.41)$$

where  $\phi_0$  denotes the starting point,  $\phi_1$  the rejected candidate and  $\phi_2$  the second stage candidate. In our context, the target distribution  $\pi(\cdot)$  is each annealing level  $p_k(\cdot)$  density distribution, the one step proposal distribution  $S_1$  is the independent approximation in equation (3.29) and the one-step acceptance probability is the global acceptance probability in (3.31). The two-step proposal density  $S_2$  can be chosen from several alternatives. In this work we use a symmetric distribution centred at the starting point  $\phi_0$ , since it can be seen as a back-guard against  $S_1$  being a deficient independent sampler [see 227, for a detailed discussion]. Therefore, the previous equation can be rewritten in compact form as

$$\alpha_{k,2}(\phi_0, \phi_2) = \min \left\{ 1, \frac{p_k(\phi_2) (1 - \alpha_k^g(\phi_1|\phi_2))}{p_k(\phi_0) (1 - \alpha_k^g(\phi_1|\phi_0))} \right\}, \quad (3.42)$$

where  $\alpha_k^g(\cdot|\cdot)$  is defined as in equation (3.31). The fact that  $S_2$  is a symmetric distribution centred in the starting point  $\phi_0$  has been used, *i.e.*  $S_2(\phi_2|\phi_0, \phi_1) = g(\phi_2|\phi_0) = g(\phi_0|\phi_2) = S_2(\phi_0|\phi_2, \phi_1)$ , where  $g(\cdot|\cdot)$  denotes such symmetric proposal density. By performing the second stage proposal, the stationary condition of  $p_k(\cdot)$  is maintained as stated in the following proposition.

**Theorem 2.** *AIMS-OPT coupled with delayed rejection in two stages leaves the target distribution  $p_k(\cdot)$  invariant at each annealing level.*

*Proof.* See Appendix 3.A at the end of this chapter for a proof using a general transition distribution  $S_2(\cdot|\cdot, \cdot)$ . ■

From the above discussion, the proposed scheme provides a fail-safe against any possible mismatch of the approximation done with (3.29). Additionally, the results presented in this dissertation correspond to the second step candidate being a Gaussian random variable,  $\xi \sim \mathcal{N}(\phi_i^{(k)}|_{c_0}\Sigma_k)$ . The ideas to accept a global transition after having accepted a local proposition can be summarised in Algorithm 3.

---

**Algorithm 3:** Global acceptance using delayed rejection

---

**if**  $\xi$  *was accepted as local candidate* **then**

    Accept  $\xi$  as a global transition with probability

$$\alpha_k^g(\xi|\phi_i^{(k)}) \quad (3.43)$$

**else**

    Generate a second candidate  $\xi_2$  from

$$\xi_2 \sim \mathcal{N}(\phi_i^{(k)}|_{c_0}\Sigma_k) \quad (3.44)$$

**if**  $\xi_2$  *is accepted with probability*  $\alpha_{k,2}(\phi_i^{(k)}, \xi_2)$  *computed as in equation (3.42)* **then**

$$\phi_{i+1}^{(k)} = \xi_2 \quad (3.45)$$

**else**

$$\phi_{i+1}^{(k)} = \phi_i^{(k)} \quad (3.46)$$

**end**

**end**

---

### 3.3 Implementation Aspects

The computational complexity of the posterior distribution in equation (3.14) is governed by the inverse of the covariance matrix  $K$  as it scales with the number of training runs  $N$ . Several solutions have been developed in the literature, such as computation of inverse products of the form  $K^{-1}u$ , with  $u \in \mathbb{R}^N$ , by means of Cholesky factors or Spectral Decomposition [see 101, for efficient implementations] to preserve numerical stability in the matrix operations [see 95]. Nonetheless, numerical stability is not likely to be achieved if the training runs are very limited, or if the sampling scheme for such training runs cannot lead to stable covariance matrices, as depicted in Figure 3.3.

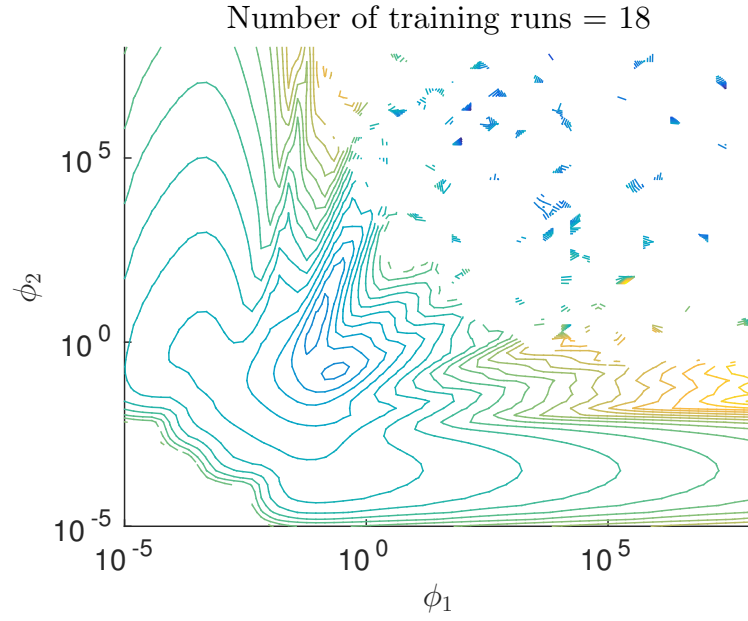
To overcome this practical deficiency, a correction term in the covariance matrix can be added in order to preserve diagonal dominance, that is, we add a *nugget* hyperparameter  $\phi_\delta$  to the covariance such that

$$K_\delta = K + \phi_\delta I, \quad (3.47)$$

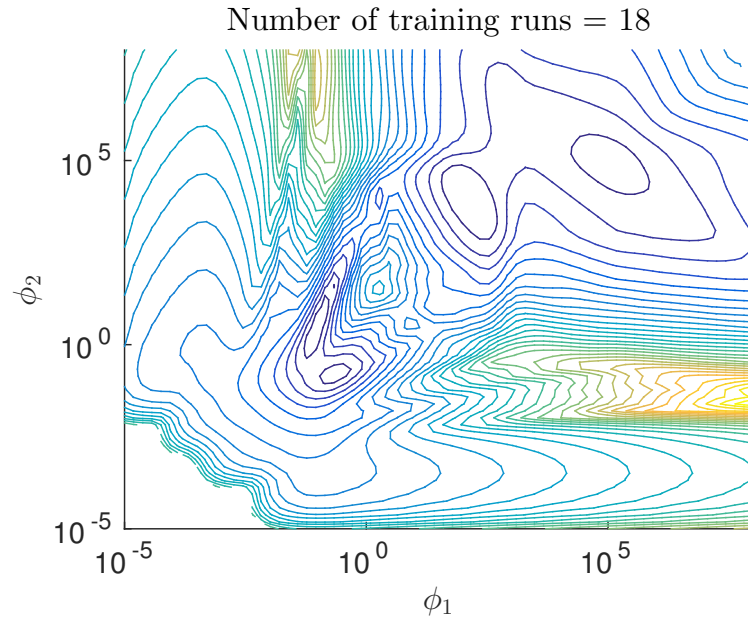
is positive definite. Doing so results in the stochastic simulator

$$y_i = \eta(\mathbf{x}_i) + \sigma^2 \phi_\delta. \quad (3.48)$$

Note that the interpolating quality of the Gaussian process is lost, however, the term  $\sigma^2 \phi_\delta$  accounts for the variability of the simulator that cannot be explained by the emulator given the original assumptions (adequacy of the covariance function, for example). The nugget can also provide further quantification of model uncertainty as it provides an alternative smoothing of an already complex surface. The nugget term also incorporates the effect of inactive inputs in the covariance kernel. Thus, it can be interpreted as a formal dimensional reduction mechanism useful in modelling complex physical processes with high-dimensional input spaces. As it is also noticed by Andrianakis and Challenor [2] and Ranjan et al. [185], the quality of the emulator changes with the inclusion of the nugget, since it modifies the objective function itself by introducing new modes in the landscape of the posterior distribution. The configuration reflected by new modes in these cases might correspond to emulators with no local dependencies and an overall simple trend, defined from the basis functions and regression hyperparameters  $\beta$ . Therefore, if a Gaussian process with no local dependencies, *e.g.* with its



(a) Numerically unstable surface



(b) Numerically stable surface

**Figure 3.3:** Projection of the negative log-posterior curves in the two dimensional length-scale space. Adding the nugget  $\phi_\delta$  results in a numerically stable surface.

mode farther away from the origin in the length-scale space, is assessed as not appropriate for the model, a regularisation term can be added in the optimisation

formulation as in [2]. This corresponds to precautions for the inclusion of the nugget and can be seen as elicited prior beliefs on the Bayesian formulation. However, by using a multi-modal sampler for stochastic optimisation as the one proposed, a robust emulator capable of mixing various possibilities can be provided. This results in an emulator that is able to cope with violations to the modelling assumptions originated by working with a limited amount of training runs.

The nugget term  $\phi_\delta$  is incorporated as a hyperparameter of the correlation function in the Bayesian inference process. As suggested by Ranjan et al. [185] a uniform prior distribution  $U(10^{-12}, 1)$  for such parameter is considered. The effect of the bounds is twofold. First, the lower bound is used to guarantee stability in the covariance matrix. Second, the upper bound is used to force the numerical noise of the simulator to be smaller than the signal noise of the emulator itself. Note that this last assumption can be omitted if the problem requires it. By considering the correlation matrix as in equation (3.47), this yields

$$\Sigma_\delta = \sigma^2 K_\delta, \quad (3.49)$$

where  $K_\delta$  denotes the corrected correlation matrix and  $\Sigma_\delta$  has been used to denote the covariance matrix of the Gaussian process. By doing so it is clear that previous considerations regarding  $\sigma^2$ , such as the ability of marginalising it as a nuisance parameter and the use of a non-informative prior remain unchanged [53].

### 3.4 Numerical Experiments using PAIMS

To illustrate the robustness of estimating the hyperparameters of a Gaussian process using the parallel AIMS-OPT framework, three test cases have been selected. The first two are common examples that can be found in the literature. The first is known as the Branin function and has been modified to resemble usual properties of engineering applications [80]. The second one [13] has been used as a two dimensional function with a challenging complexity for emulating purposes. The third example presented in this section comes from a real dataset also presented in Bastos and O'Hagan [13]. In all the examples it is assumed that  $h(\mathbf{x}) = (1, x_1, \dots, x_p)^T$ . Regarding the nugget, a sigmoid transformation has been performed in order to sample from a Gaussian distribution. Namely, we sample an auxiliary  $z_\delta$  as part of the multivariate Gaussian in (3.37), and compute the



nugget as

$$\theta_\delta = \frac{1 - l_b}{1 + \exp(-z_\delta)} + l_b, \quad (3.50)$$

where  $l_b$  is the lower bound for the nugget, which is set equal to  $10^{-12}$ . Additionally, the uniform meta-prior distribution of equation (3.22) has been considered in a practical support of the length-scale parameters in the logarithmic space, namely a uniform distribution with support in  $[-7, 7]$ . For the nugget, a truncated beta distribution with parameters  $\alpha = \beta = 0.5$  has been considered since it corresponds to a non informative meta-prior in the interval  $[l_b, 1]$ . Here the prefix *meta* has been used to refer to the algorithm's prior distribution and to set a clear distinction from the prior used in the modelling assumptions in equation (3.13).

The code has been implemented in MATLAB and all examples have been run in a GNU/Linux machine with an Intel i5 processor with 8 Gb of RAM. For the purpose of reproducibility, the code used to generate the examples is available for download at [https://github.com/agarbuno/paims\\_codes](https://github.com/agarbuno/paims_codes).

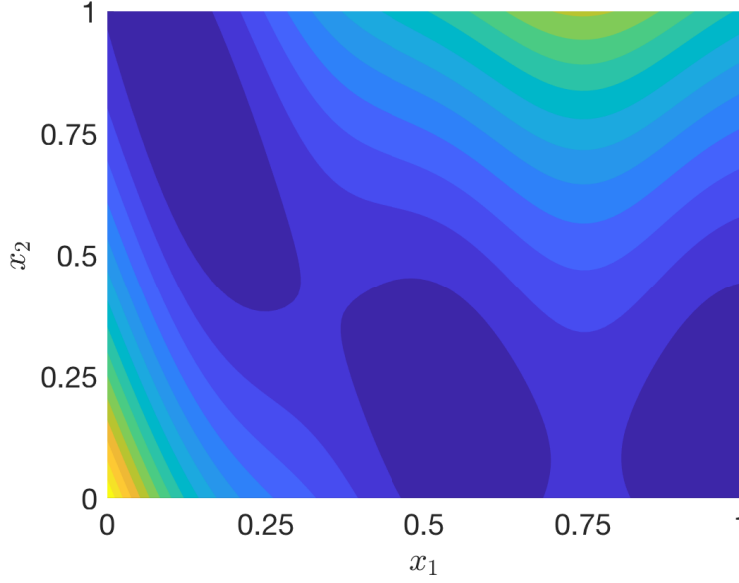
### 3.4.1 Branin Function

The version of the Branin function used in this dissertation is a modification made by Forrester et al. [80] for the purpose of Kriging prediction in engineering applications. It is a rescaled version of the original in order to bound the inputs to the rectangle  $[0, 1] \times [0, 1]$ , with an additional term that modifies its landscape to include a global optimum. Namely,

$$f(\mathbf{x}) = \left( \bar{x}_2 - \frac{5.1}{4\pi^2} \bar{x}_1^2 + \frac{5}{\pi} \bar{x}_1 - 6 \right)^2 + 10 \left[ \left( 1 - \frac{1}{8\pi} \right) \cos(\bar{x}_1) + 1 \right] + 5\bar{x}_1, \quad (3.51)$$

where  $\bar{x}_1 = 15x_1 - 5$  and  $\bar{x}_2 = 15x_2$ . For completeness, Figure 3.4 shows the contour levels of the Branin function in the domain of interest  $\mathcal{X} = \{(x_1, x_2) \in [0, 1] \times [0, 1]\}$ .

For this case, a sample of 18 design points were chosen with a Latin hypercube sampling (LHS) scheme for training. Additional 25 points were chosen from an independent LHS for validation purposes. The resulting log-posterior function possesses 4 different modes in its landscape (see Figure 3.5a) leading to 4 possible configurations of the correlation function. Thus, the impact of the training runs used to construct the emulator is evident. Among these modes, 4 different types



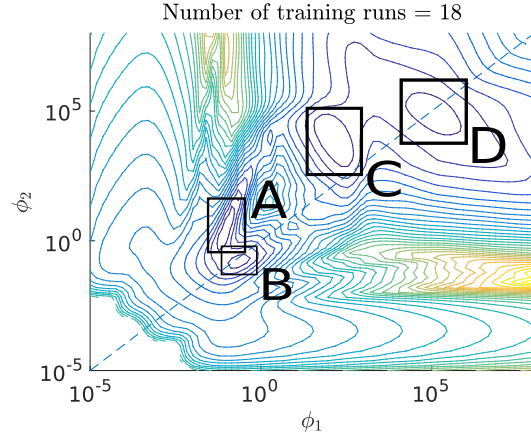
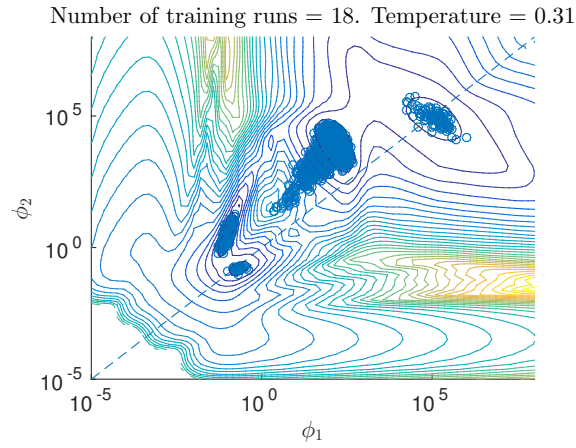
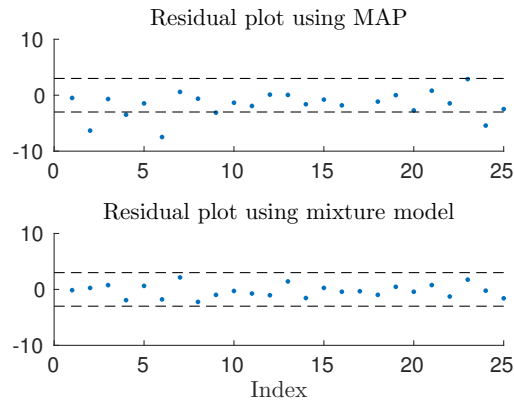
**Figure 3.4:** *Contour levels of Branin function*

of emulators can be distinguished: an emulator with high sensitivity to changes in input  $x_1$  (mode A in Figure 3.5a); an emulator with rapid changes in  $x_2$  for the correlation structure of the training runs (mode B); a limiting case where dimension  $x_2$  is disregarded in the correlation function, due to a high value in  $\phi_2$  (mode C); or a second limiting emulator which approximates a Bayesian linear regression model (mode D) [see 2, for a detailed discussion].

For this example, two thousand samples were generated in each annealing level. The parallel AIMS-OPT algorithm generated 7 annealing levels to produce the samples in Figure 3.5b. The RMSE of the MAP model is 7.068 whereas the RMSE of the mixture is 15.099 which is an indication that in terms of brute prediction, the mixture model could be improved by taking more samples. Figure 3.5c depicts the standardised residuals from both the MAP approach (top) and the mixture model (bottom) using equations (3.10) and (3.11) with uniform weights in the sample. The standardised residuals are defined as

$$r(\mathbf{x}) = \frac{y(\mathbf{x}) - \mu(\mathbf{x})}{\sqrt{\sigma^2(\mathbf{x})}}, \quad (3.52)$$

where  $y$  is the output for configuration  $\mathbf{x}$ ,  $\mu(\mathbf{x}) = E[y|\mathbf{x}, \mathcal{D}]$  and  $\sigma^2(\mathbf{x}) = \text{var}(y|\mathbf{x}, \mathcal{D})$ , the posterior mean and variance for configuration  $\mathbf{x}$  [see 13, for

(a) *Level curves and modes*(b) *Parallel AIMS-OPT sample*(c) *Residual plots*

**Figure 3.5:** *Projection of the negative log-posterior curves in the two dimensional length-scale space for the Branin simulator. The minimum possible value of  $10^{-12}$  for the nugget  $\phi_\delta$  has been used for such projection. The reference diagonal helps visualise the regions where the length scales favour one dimension over the other.*

a more detailed discussion on diagnostics]. By marginalising the hyperparameters it is clear that our estimation is a more robust in terms of error prediction. Even with such limited amount of information the residuals suggest that the uncertainty is being incorporated appropriately in the marginalised predictive posterior distribution in equation (3.6). The standardised residuals are inside the 95% confidence bands, assuming approximate normality, though not too close to 0. This is an indicator that although greater variability is expected, excessively large variances are avoided. This is done by means of the integrated predictive distribution and the use of the proposed sampler to build a mixture of emulators leaving the predicted errors inside appropriate bounds.

### 3.4.2 2D Model

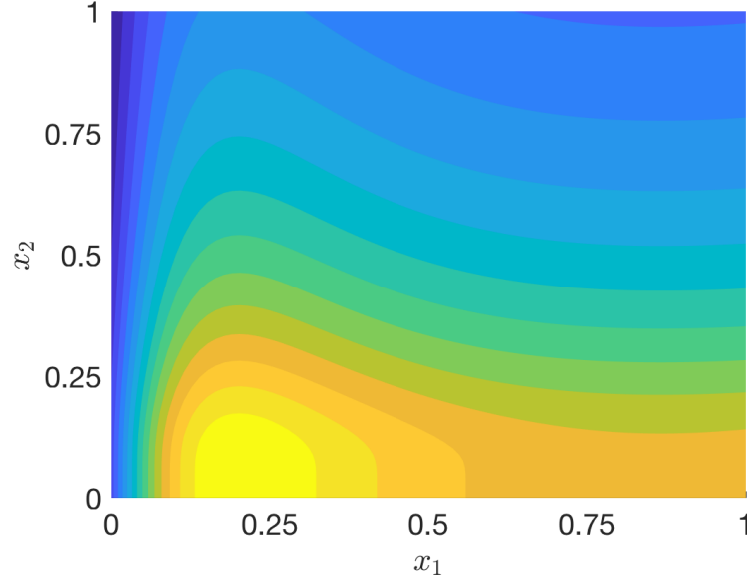
This function has already been used as an example for emulation purposes and can be found in GEM-SA software web page (<http://ctcd.group.shef.ac.uk/gem.html>). Even though it is a two dimensional problem it also serves as a good illustration of the importance of estimating the hyperparameters of a Gaussian process with a multi-modal sampler. The mathematical expression for this simulator is

$$f(\mathbf{x}) = \left[ 1 - \exp \left( -\frac{0.5}{x_2} \right) \right] \left( \frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^2 + 500x_1^2 + 4x_1 + 20} \right). \quad (3.53)$$

For completeness, Figure 3.6 depicts the contour levels in the domain of interest  $\mathcal{X} = \{(x_1, x_2) \in [0, 1] \times [0, 1]\}$  for this 2D model.

As in the previous case, the training runs and the modelling assumptions fail to summarise the uncertainty in a uni-modal posterior distribution. As in the Branin example, 18 design points and 25 validation points were selected using two independent LHS designs in the rectangle  $[0, 1] \times [0, 1]$ . It can be seen from Figure 3.7a that the modes are separated by a wide valley of low posterior probability, which can become an overwhelming task for traditional MCMC samplers. The proposed sampler is able to cope with all local and global spread dynamics present in the neighbourhoods of the modes it encounters, as shown in Figure 3.8a.

Depicted in Figures 3.7a and 3.7b the use of the reference prior in the posterior distribution removes probability mass from the neighbourhood around the origin. This validates the use of the reference prior to cut out regions from the space of

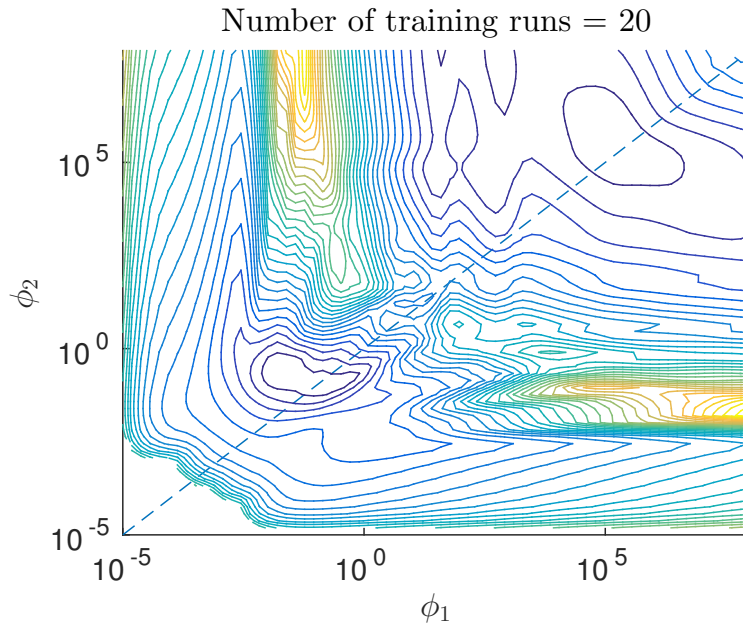


**Figure 3.6:** *Contour levels of the 2D Model*

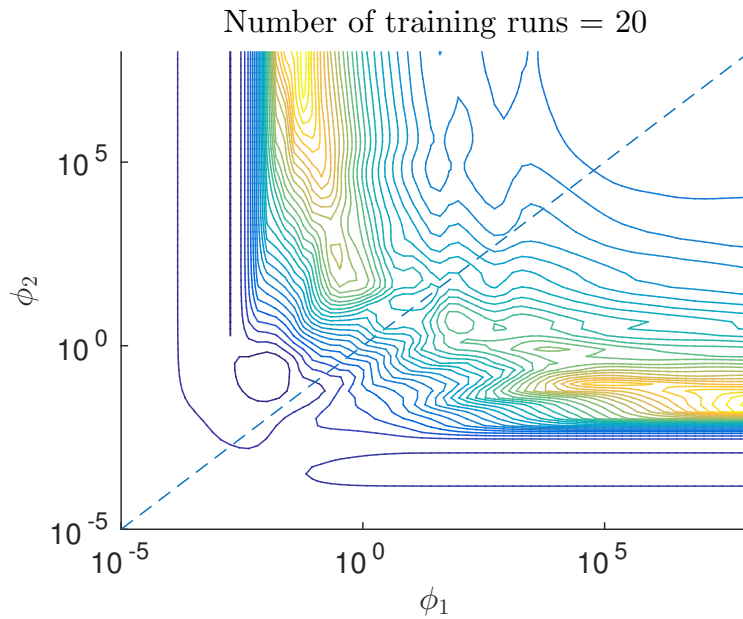
hyperparameters for the sampling and exploit the most information contained in the data available, namely, the training runs  $\mathcal{D}$ . As in the previous example, two thousand samples were generated in each annealing level. The parallel AIMS-OPT algorithm generated 7 annealing levels to produce the samples in Figure 3.8a. In terms of prediction accuracy, we now obtain that the RMSE is 1.356 for the MAP estimate and 1.345 for the mixture model. While as for the residuals, we can see from Figure 3.8b that the mixture model allows for a more robust prediction of the error, by means of increasing the variability in particular locations. This can be seen as the standardised residuals are concentrated within the 95% confidence bands of an approximate assumed normality, resulting in a more robust estimation of the error by the use of a mixture model. This motivates the use of multi-modal density samplers in the context of optimisation, where if a single candidate is provided the overall error prediction of the emulator might be biased towards more concentrated predictions around the mean estimation.

### 3.4.3 Nilson-Kuusk Model

Although the focus of this dissertation is engineering models, the framework developed in this chapter is, naturally, not exclusive to engineering simulations.

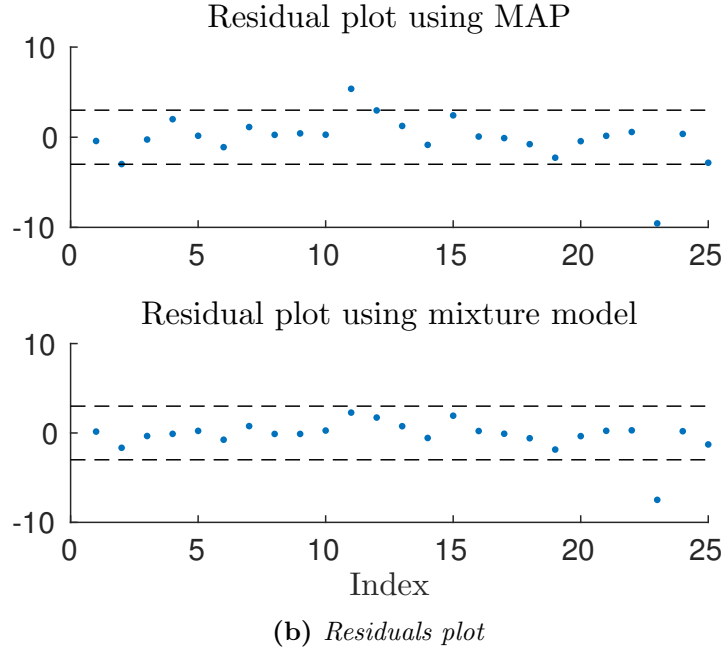
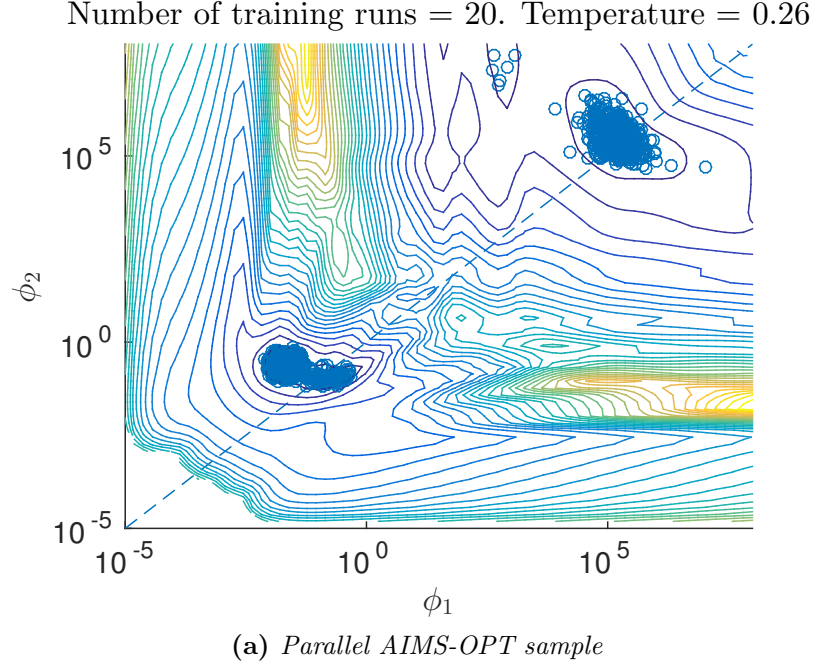


(a) Level curves with reference prior



(b) Level curves with uniform prior

**Figure 3.7:** Projection of the negative log-posterior curves in the two dimensional length-scale space for the 2D Model simulator. The minimum possible value of  $10^{-12}$  for the nugget  $\phi_\delta$  has been used for such projection. The reference diagonal helps visualise the regions where the length scales favour one dimension over the other.



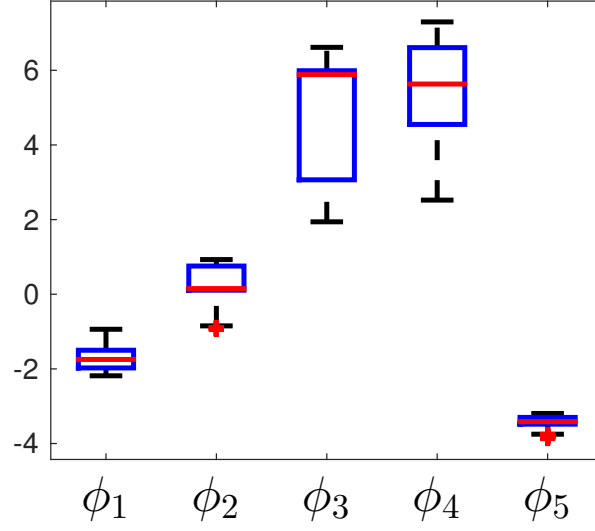
**Figure 3.8:** Projection of the negative log-posterior curves in the two dimensional length-scale space for the 2D Model simulator. In the upper panel, dots depict the posterior sample points generated with the PAIMS algorithm. The lower panel shows diagnostics in the validation data using the MAP estimate and the posterior samples generated by PAIMS.

For this experiment, the simulator is the Nilson-Kuusk model for the reflectance for a homogeneous plant canopy. Such model is a five dimensional simulator whose inputs are the solar zenith angle, the leaf area index, relative leaf size, the Markov clumping parameter and a model parameter  $\lambda$  [see 168, for further details on the model itself and the meaning of the inputs and outputs]. For the analysis presented in this dissertation a single output emulator is assumed and the set of the inputs have been rescaled to fit the hyper-rectangle  $[0, 1]^5$  on a five dimensional space as in Bastos and O’Hagan [13].

As in the previous test cases, the design points were chosen by Latin hypercube designs (100 for this case). In this example, the dimension of the problem makes it impossible to plot the level curves of the posterior distribution for the length scale hyperparameters to visualize potential multiple modes. However, the samples can be visualized by means of a box-plot as shown in Figure 3.9, where the red line denotes the median, the edges of the box the 25<sup>th</sup> and 75<sup>th</sup> percentiles, and the whiskers cover the most extreme cases. The samples are obtained after completing 10 levels of the parallel AIMS-OPT algorithm. The box-plots of the approximate optimal solutions strongly suggest that the samples come from a multi-modal posterior distribution. This can be seen from the location of the edges of the boxes and the median for any given input. The last input possesses a very limited spread which might denote a high concentration around one mode. Note that as the magnitude of the length-scale increases, thus reducing the sensitivity of the simulator to such input, the length-scales are located in what can be seen as either a plateau or regions of modes with negligible difference in the posterior density. Additionally, from the range of values that are covered in log-space, it can be noted that the output of the simulator appears to be insensitive to changes of the third and fourth input. Furthermore, a limit-case emulator can be suggested by the boxplot in Figure 3.9 by considering a surrogate with no third and fourth inputs in the model. Notice the scales for such hyperparameters in logarithmic space.

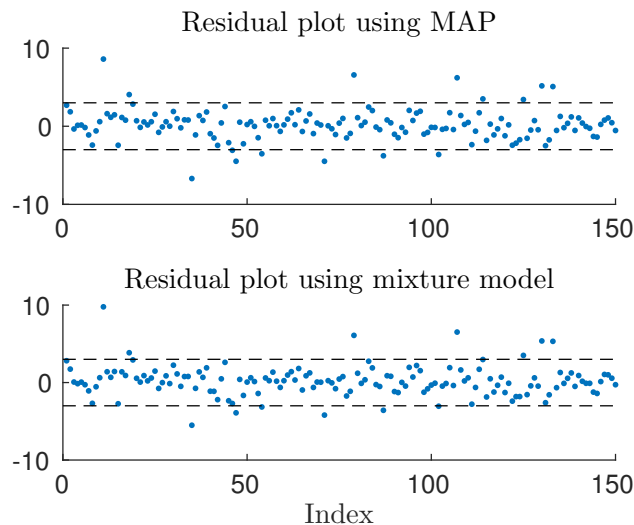
Due to the larger number of dimensions, five thousand samples were generated for each annealing level. In this case we have that the RMSE of the MAP estimate is 0.022 while the RMSE of the mixture proposal is 0.021 which is a consequence of the posterior distribution being highly concentrated around one mode, in a particular set of length-scales ( $\phi_1, \phi_2$  and  $\phi_5$ ) while being less specialised for the less sensitive ones ( $\phi_3$  and  $\phi_4$ ). The values for  $\phi_3$  and  $\phi_4$  are so large that a linear





**Figure 3.9:** Box-plots of the sample of length-scales obtained by the parallel asymptotically independent Markov sampling.

dependence of the simulator output with respect to both parameters is suspected. In Figure 3.10 there is evidence that even with such behaviour the predictive error is improved by narrowing the spread of the standardised residuals, as before, a consequence of an increased estimation of the variability in particular locations. In this case the residuals cannot all be contained in the approximate normality 95% bands but as noted by Bastos and O’Hagan [13] in their experiments there is strong evidence that more runs of the simulator are needed to adequately build a statistical surrogate. Due to the highly concentrated posterior density around the high sensitive length-scales there seems to be no apparent gain from using the mixture model. However, it can be noted from Figure 3.9 that by acknowledging the variability of the hyperparameters, a better understanding of the sensitivity of the simulator with respect to the inputs is achieved. An improved and more robust uncertainty analysis of the simulator can be provided in this case understanding the wide spread of length-scales for particular dimensions. For instance if screening is performed, the MAP estimate will fail to summarise the wide posterior density with respect to  $\phi_3$  and  $\phi_4$  and this in turn, will provide partial information. This analysis cannot be performed solely by maximising the posterior density. Therefore the proposed method provides additional insight of the sensitivity of both simulator and emulator.



**Figure 3.10:** *Residuals plot diagnostic for the Nilson-Kuusk simulator.*

### 3.5 Discussion on PAIMS Results

In the previous sections, a new sampler based on the Asymptotically Independent Markov Sampling (AIMS) method is proposed. The main objective being the estimation of the hyperparameters of a Gaussian process. The AIMS-OPT algorithm, used in stochastic optimisation, provides a robust computation of the MAP estimates of the hyperparameters. This is done by providing a set of approximations to the optimal solution instead of a single approximation as it is so frequently done in the literature. The problem is approached in a combined effort from the computational, optimisation and probabilistic perspectives which serve as solid foundations for building surrogate models for computationally expensive computer codes.

The original AIMS algorithm has been extended to provide an efficient sampling alternative in computational terms, by means of parallelisation, as well as an effective sampler with good mixing qualities, by means of both the delayed rejection and adaptive modification exposed. It has been demonstrated that by using the parallel AIMS-OPT (PAIMS-OPT) algorithm it is possible to acknowledge uncertainty in the structure of the emulator proposed as illustrated in the examples provided. Structural uncertainty should be taken into account to determine when the training runs available are sufficient to narrow the posterior distribution of the hyperparameters to a uni-modal convex distribution. Even though it has been

proven to be effective in lower and medium dimensional design spaces, research in high dimensional spaces has been left for future research.

## 3.6 Conclusion

This chapter discussed the use of Gaussian processes as emulators of computationally expensive simulators such as the ones encountered in engineering applications. The Gaussian process emulator was formulated in a full Bayesian setting. This chapter also introduced the proposed PAIMS algorithm to perform the marginalisation of the hyperparameters. The algorithm's flexibility allows also for an optimisation strategy as well. This is known as subset optimisation and it enables the identification of the modes of the posterior distribution. The efficiency of the generation of samples was achieved by coupling adaptive scaling for each intermediate level in the AIMS framework, and also by incorporating a delayed-rejection mechanism to improve the acceptance rate without compromising the exploration of the space. In the following chapter, the delayed rejection ideas are explored further through the use of slice sampling and are incorporated into the AIMS framework. This leads to the development of the Transitional Annealed Adaptive Slice Sampling (TA<sup>2</sup>S<sup>2</sup>) in Chapter 4, which will be later applied in the history matching setting in Chapter 5 under both modes of operations. That is, sampling for the hyperparameters of Gaussian process emulator, and subset optimisation to lead the refocusing in history matching.

### 3.A Appendix

This appendix shows that using the delayed rejection algorithm in the AIMS framework leaves the target distribution  $p_k(\cdot)$  invariant.

A sufficient condition to prove that indeed  $p_k(\cdot)$  is the stationary distribution for the Markov chain is to prove that the detailed balance condition is satisfied. Since the first stage approval has been proven to satisfy the detailed balance condition in Zuev and Beck [226], it will only be proved for the second stage sampling.

Let  $f_k(\phi_2|\phi_0)$  describe the AIMS-OPT delayed transitions in the  $k$ -th annealing level from  $\phi_0 \rightarrow \phi_2$ , with  $\phi_2 \neq \phi_0$ . Let  $\phi_1$  be the rejected transition in the first stage, for any  $\phi_0, \phi_1, \phi_2 \in \Phi \setminus \{\phi_1^{(k-1)}, \dots, \phi_n^{(k-1)}\}$ . It will be proved that for such candidates the following holds:

$$p_k(\phi_0)f_2(\phi_2|\phi_0) = p_k(\phi_2)f_2(\phi_0|\phi_2). \quad (3.54)$$

As seen from the description in section 3.2.4 it follows that

$$f_k(\phi_2|\phi_0) = \underbrace{\hat{p}_{k,n}(\phi_1)}_{\text{generate } \phi_1} \underbrace{(1 - a_1(\phi_0, \phi_1))}_{\text{reject } \phi_1} \underbrace{S_2(\phi_2|\phi_0, \phi_1)}_{\text{generate } \phi_2} \underbrace{a_2(\phi_0, \phi_2)}_{\text{accept } \phi_2}, \quad (3.55)$$

where it is used the fact that AIMS-OPT generates first stage proposals with an independent approximate distribution. Recall that the probability of a second stage proposal is

$$a_2(\phi_0, \phi_2) = \min \left\{ 1, \frac{p_k(\phi_2) S_2(\phi_0|\phi_2, \phi_1) (1 - a_1(\phi_2, \phi_1))}{p_k(\phi_0) S_2(\phi_2|\phi_0, \phi_1) (1 - a_1(\phi_0, \phi_1))} \right\} \quad (3.56)$$

and the fact that for any two positive numbers  $a, b$  the equality  $a \min\{1, b/a\} = b \min\{1, a/b\}$  is satisfied. With these two equalities we can substitute the left hand side of equation (3.54) as

$$\begin{aligned} p_k(\phi_0)f_2(\phi_2|\phi_0) &= \hat{p}_{k,n}(\phi_1) [p_k(\phi_0) S_2(\phi_2|\phi_0, \phi_1) (1 - a_1(\phi_0, \phi_1))] a_2(\phi_0, \phi_2) \\ &= \hat{p}_{k,n}(\phi_1) [p_k(\phi_2) S_2(\phi_0|\phi_2, \phi_1) (1 - a_1(\phi_2, \phi_1))] a_2(\phi_2, \phi_0) \\ &= p_k(\phi_2) f_2(\phi_0|\phi_2), \end{aligned} \quad (3.57)$$

which proves the detailed balance for the second stage proposal. Note that the proof has been made with no further assumptions about the second stage proposal distribution  $S_2(\phi_2|\phi_0, \phi_1)$ , as it can be defined from several candidates.

In this work, a symmetric proposal that ignores the rejected sample has been used since it can be interpreted as a Random Walk safeguard against a possible ill approximation done by the independent sampler.



## CHAPTER 4

---

### Gaussian Process Hyperparameters: Transitional Annealed Adaptive Markov Sampling<sup>1</sup>

---

The previous chapter proposed a new algorithm to sample the hyperparameters of a Gaussian process. This chapter, introduces a new algorithm based on this work given that the concept of delayed rejection can be further strengthened. This is done by incorporating Slice Sampling [165] in the proposal distribution. Overall, this proposed sampling scheme is based on two principles and can be used for multi-modal distributions. The first principle is to use the concept of *crumb* introduced by Neal [165] for a multivariate adaptive slice sampler to guide the proposal distribution. The second principle is based on the AIMS framework introduced by Zuev and Beck [226] and discussed in Chapter 3. This in turn enables the generation of samples through a sequence of nested subsets as in Stochastic Subset Optimisation [209, 210]. The use of delayed rejection in the proposed asymptotically independent Markov sampler [85] of the previous chapter, has proven to enhance the mixing capabilities in highly correlated probability models. To the author’s knowledge, coupling the adaptive slice sampling algorithm with a sequential sampler has not been explored previously. This presents an opportunity to develop efficient sampling algorithms for multi-modal distributions.

---

<sup>1</sup>The algorithms and ideas presented in this chapter have been published in Garbuno-Inigo et al. [84]

The main advantage of the proposed scheme is that it requires little tuning of parameters as it automatically learns the sequence of temperatures for an annealing schedule, as opposed to being tuned by trial and error [31]. The samples generated at any previous annealing level can be exploited further as they provide the crumbs needed for the sampling in the next annealing level. Additionally, embedding the sampler with the Transitional Markov chain Monte Carlo method [45] results in an algorithm that can be run in a cluster of cores, if available. By using the proposed Transitional Annealed Adaptive Slice Sampling algorithm ( $TA^2S^2$ ) to sample the hyperparameters of a Gaussian process, the resulting emulator is built taking into account both a probabilistic and computationally efficient perspective. Efficiency is gained as the sampler adapts the proposal distribution, which for other MCMC schemes is a highly sensitive parameter to be tuned. The probabilistic strategy to treat the problem in a Bayesian manner accounts for the uncertainty that stems from the unknown parameters. This adds a layer of structural uncertainty to the model. Additionally, model uncertainty is mitigated by adding numerical stabilisation measures in the Gaussian process model as in [185, 2] in a fully Bayesian framework.

This chapter is organised as follows. In Section 4.1 the concepts of Slice Sampling and Adaptive Slice Sampling are presented. Section 4.2 presents the proposed  $TA^2S^2$  algorithm, as well as extensions needed for a parallel implementation. In Section 4.3, some illustrative examples are used to discuss the efficiency and robustness of  $TA^2S^2$ . Section 4.4 includes a discussion of such results. Section 4.5 presents an industrial application of the proposed sampler to train a Gaussian process emulator. Finally, Section 4.6 presents concluding remarks for the chapter.

## 4.1 Slice Sampling

The slice sampling algorithm [165] is a method to simulate a Markov chain of a random variable  $\boldsymbol{\theta} \in \Theta$ . This is done by introducing an auxiliary random variable  $u \in \mathcal{U} \subseteq \mathbb{R}$  and sampling from the joint distribution on the extended space  $\Theta \times \mathcal{U}$ . The marginal of  $\boldsymbol{\theta}$  is recovered by disregarding the values of  $u$  in the Markov chain, a consequence of defining an appropriate conditional distribution for  $u$ , given  $\boldsymbol{\theta}$ . The samples are generated by an iterative Gibbs sampling schedule to recover pairs  $\{(\boldsymbol{\theta}_i, u_i)\}_{i=1}^N$ , which follow the joint density probability distribution



$$\pi(\boldsymbol{\theta}, u) \propto I_{\{u < \pi(\boldsymbol{\theta})\}}(\boldsymbol{\theta}, u), \quad (4.1)$$

where  $I_E(\cdot)$  is the indicator function for the set  $E \subset \Theta \times \mathcal{U}$ , and  $\pi(\boldsymbol{\theta})$  is the target distribution of  $\boldsymbol{\theta}$ . Slice Sampling first generates  $u$  from the conditional distribution of  $u \mid \boldsymbol{\theta}$  specified as a uniform on the interval  $(0, \pi(\boldsymbol{\theta}))$ . It then samples  $\boldsymbol{\theta}$ , conditioned in  $u$  from a uniform distribution in the *slice* defined by the set

$$S_u = \{\boldsymbol{\theta} : u < \pi(\boldsymbol{\theta})\}. \quad (4.2)$$

Since the marginal satisfies  $\int_0^{\pi(\boldsymbol{\theta})} \pi(\boldsymbol{\theta}, u) du = \pi(\boldsymbol{\theta})$ , samples from the target distribution can be recovered by disregarding the auxiliary component of the joint samples. If the target distribution is a non-normalised probability density  $f(\boldsymbol{\theta})$  then the joint distribution can be written as

$$\pi(\boldsymbol{\theta}, u) = \frac{1}{Z} I_{\{u < f(\boldsymbol{\theta})\}}(\boldsymbol{\theta}, u), \quad (4.3)$$

where  $Z = \int_{\Theta} f(\boldsymbol{\theta}) d\boldsymbol{\theta}$  and the previous considerations for the marginal of  $\boldsymbol{\theta}$  follow. In the context of Gaussian processes it should be noted that floating-point underflows are common due to ill-conditioning of the matrix  $K$  in equation (3.14). Thus, in order to compute stable evaluations of the target distribution in Slice Sampling, it is preferable to evaluate the negative logarithm of the target density. In such case, equation (4.2) can be computed as stated in the following proposition.

**Theorem 3** (Slice characterisation). *Given the state of the Markov chain  $\boldsymbol{\theta}_0$ , the uniform distribution for the next candidate has support in the slice given by*

$$S_{\boldsymbol{\theta}_0} = \{\boldsymbol{\theta} : z > \mathcal{H}(\boldsymbol{\theta})\}, \quad (4.4)$$

where  $\mathcal{H}(\cdot)$  denotes the negative logarithm of the target density and  $z = \mathcal{H}(\boldsymbol{\theta}_0) + e$ , with  $e$  distributed as an exponential random variable with mean equal to 1.

*Proof.* The result follows from the fact that for a given state  $\boldsymbol{\theta}_0$  of the Markov chain, the auxiliary uniform random variable defining the slice can be written as the product  $u \times f(\boldsymbol{\theta}_0)$  with  $u$  uniformly distributed in the interval  $(0, 1)$ . Thus, the slice is defined as

$$\begin{aligned}
 S_{\theta_0} &= \{\boldsymbol{\theta} : u f(\boldsymbol{\theta}_0) < f(\boldsymbol{\theta})\} \\
 &= \{\boldsymbol{\theta} : -\log(f(\boldsymbol{\theta}_0)) - \log(u) > -\log(f(\boldsymbol{\theta}))\} \\
 &= \{\boldsymbol{\theta} : \mathcal{H}(\boldsymbol{\theta}_0) + e > \mathcal{H}(\boldsymbol{\theta})\},
 \end{aligned} \tag{4.5}$$

where it is easy to prove that  $e = -\log(u)$  is distributed as an exponential random variable with mean 1 and  $\mathcal{H}(\cdot)$  denotes the negative logarithm of the target density. ■

The main concern when implementing Slice Sampling is the ability to sample uniformly from the slice. In one-dimensional applications, the slice can be defined in many ways. The canonical example is a stepping-out and shrinkage procedure which aims to adapt an initial interval centred in the current state of the Markov chain [see 165, for further details].

### 4.1.1 Adaptive Slice Sampling

For multivariate distributions, the concept of the slice extends naturally. However, methods based on intervals (*e.g.* the stepping-out and shrinking procedure) become dramatically slow as the dimension of the problem increases. This is due to the generalisation of intervals as hyper-rectangles in  $\mathbb{R}^p$  and the need to compute the target function for each vertex a repeated number of times along the expansion and shrinkage of the boundaries. For Gaussian process emulators, the task of evaluating the target density becomes expensive, a consequence of the non-parametric nature of the model and the computational cost of evaluating equation (3.14). If multiple evaluations are needed for the construction of the Markov chain either because of a high rejection rate, difficult characterisation of the slice or if longer chains are required, simulation by MCMC with slice sampling becomes computationally expensive and inefficient. Therefore, other alternatives are preferable.

This work employs a framework proposed by [165] for adaptive slice sampling in multivariate applications. The key idea is the use of the information provided by the rejected samples in order to lead the future generation of a candidate inside the slice. In this framework, the evidence gathered by the rejected candidates is referred to as *crumbs*, as they will be “followed” towards the slice.

## 4.2 Transitional Annealed Adaptive Slice Sampling

As previously stated, in order to marginalise the posterior predictive distribution in equation (3.14), Monte Carlo integration is usually performed when aiming at a fully Bayesian treatment of Gaussian process surrogates. This is usually done by Hybrid Monte Carlo [163, 222] which is capable of suppressing the Random Walk behaviour of traditional MCMC methods. Nonetheless, the tuning of this kind of algorithm is problem-dependent and expert knowledge is crucial for an optimal sampling schedule. The development of Elliptical Slice Sampling [159] provides a framework for the simulation of the hyperparameters of a Gaussian process with little tuning required from the analyst [158]. However, this is only applicable when the posterior predictive distribution for the hyperparameters is of the form

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto \mathcal{N}(l(\boldsymbol{\theta})|\boldsymbol{\mu}, \Sigma) p(\boldsymbol{\theta}), \quad (4.6)$$

where  $p(\boldsymbol{\theta})$  denotes the prior distribution,  $\mathcal{N}(\cdot|\cdot, \cdot)$  is a Gaussian distribution and  $l(\cdot)$  is a latent variable that depends on the hyperparameters. As it can be seen from the integrated posterior in equation (3.14), this is not the expression for the assumed posterior distribution. The difference stems from  $\sigma$  being considered a nuisance parameter and the prior considered for the length-scales of the Gaussian process.

This work proposes the Transitional Annealed Adaptive Slice Sampling algorithm (TA<sup>2</sup>S<sup>2</sup>), which can also be used in other applications of Bayesian inference and Stochastic optimisation. The sampling problem is formulated based on the Asymptotically Independent Markov Sampling [14] framework. This treats the sampling problem as a simulated annealing algorithm. The objective is to sample from intermediate posterior distributions  $p_k(\phi|\mathcal{D})$  that eventually converge to the true posterior. This is done by tempering the posterior distribution by means of a monotonically decreasing sequence of temperatures  $\tau_k$  converging to 1. Let  $\{p_k(\phi|\mathcal{D})\}_{k=1}^{\infty}$  be the sequence of density distributions in the annealing schedule such that

$$p_k(\phi|\mathcal{D}) \propto p(\phi|\mathcal{D})^{1/\tau_k} = \exp\{-\mathcal{H}(\phi|\mathcal{D})/\tau_k\}, \quad (4.7)$$

where  $\mathcal{H}(\phi|\mathcal{D})$  denotes the negative integrated log-posterior distribution of the

length-scale hyperparameters, given the set of training runs  $\mathcal{D}$ .

The algorithm provides a sequence of nested subsets  $\Phi_{k+1} \subseteq \Phi_k$  converging to the set of posterior samples denoted by  $\Phi^*$ . The temperature is learned through an automatic mechanism to determine the sequence of distributions. By construction, the sample in the first level of annealing is distributed uniformly on a *practical support* of the sampling space [see 126, for more a detailed discussion]. For the limiting case, the samples are uniformly distributed in the support of the posterior density. Both these observations can be summarised by

$$\lim_{\tau \rightarrow \infty} p_\tau(\phi|\mathcal{D}) = U_\Phi(\phi), \quad (4.8)$$

$$\lim_{\tau \rightarrow 1} p_\tau(\phi|\mathcal{D}) = U_{\Phi^*}(\phi), \quad (4.9)$$

where  $U_A(\phi)$  denotes a uniform distribution over the set  $A$  for every  $\phi \in A$ .

### 4.2.1 Annealing at Level $k$

This subsection focuses on the sampling carried out by  $TA^2S^2$  at the  $k$ -th level of the annealing sequence. It is assumed that a sample from level  $k-1$ , which is distributed according to  $p_{k-1}(\phi|\mathcal{D})$ , has already been generated. Let  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)}$  denote such sample and let  $N$  be the sample size in each annealing level. Following the ideas discussed in Section 4.1.1 for Adaptive Slice Sampling, the *crumb* formulation will be exploited. The samples from the previous level play the role as the crumbs to be followed to generate candidates from each slice. Thus, retaining information from the posterior landscape and limiting the amount of evaluations of the integrated posterior, which can be expensive for a reasonable number of training runs. Firstly, note that Proposition 1 implies the following

**Theorem 4** (Slice set at the  $k$ -th level). *The slice defined in the  $k$ -th annealing level, given the current state of the Markov chain  $\phi_0$ , is given by*

$$S_{\phi_0}^k = \{\phi : z_k > \mathcal{H}(\phi|\mathcal{D})\}, \quad (4.10)$$

where  $z_k = \mathcal{H}(\phi_0|\mathcal{D}) + e_k$ , with  $e_k$  an exponential random variable with mean  $\tau_k$ .

As in other Sequential Monte Carlo algorithms [56, 77], let us define the importance weights of the samples  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)}$  as

$$\omega_j^{(k-1)} = \frac{p_k(\phi_j^{(k-1)})}{p_{k-1}(\phi_j^{(k-1)})} \propto \exp \left\{ -\mathcal{H}(\phi_j^{(k-1)} | \mathcal{D}) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\}, \quad (4.11)$$

$$\bar{\omega}_j^{(k-1)} = \frac{\omega_j^{(k-1)}}{\sum_{j=1}^N \omega_j^{(k-1)}}, \quad (4.12)$$

where  $\omega_j^{(k-1)}$  denotes the importance weights and  $\bar{\omega}_j^{(k-1)}$  the normalised importance weights. The weights allow one to measure the importance of each sample as being drawn for the next annealing level.

The proposal for a new state of the Markov chain, given the current one  $\phi_0$ , is generated as follows. A slice is obtained as in equation (4.10) by generating an exponential random variable with mean  $\tau_k$ , thus defining the slice  $S_{\phi_0}^k$  for the current state. A first crumb is randomly selected from the set of past approximations that lie inside the slice. This means selecting a uniformly distributed index  $j$  from the set

$$\mathcal{J} = \left\{ j \in \{1, \dots, N\} : \phi_j^{(k-1)} \in S_{\phi_0}^k \right\}. \quad (4.13)$$

The points  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)}$  are uniformly distributed in the approximation set  $\Phi_{k-1}$  and will be used as markers for the annealing level  $k$ . If the above index set is empty, there is evidence of the annealing temperature being decreased too rapidly. A fail-safe can be used by generating a crumb from a wide Gaussian distribution centred at the current state of the Markov chain. This is also known as Defensive Sampling. It is applied to importance samplers to improve their robustness to explore the complete sampling space [112]. Additionally, as it is done in other Sequential Monte Carlo methods [57], a renewal component can be added. The renewal is performed as the crumbs are selected from the markers, due to the fact that relying on the sample from the previous level can lead to bias in the simulations. To this end, if the index set  $\mathcal{J}$  is empty, or with probability  $p_{\text{renew}}$ , the crumb  $\varsigma_1$  will be distributed as

$$\varsigma_1 \sim \mathcal{N}(\phi_0, c_0^2 \Sigma_k), \quad (4.14)$$

where  $c_0$  is a spread parameter associated with the annealing sequence, and  $\Sigma_k$  denotes a covariance matrix at level  $k$ . Typical choices for the covariance matrix

are the identity matrix  $I_{p \times p}$  or a diagonal matrix  $\text{diag}\{d_1, \dots, d_p\}$  which defines a different scale for each variable. In order to use a better proposal in terms of scales and correlations observed along the annealing sequence, we define  $\Sigma_k$  as the weighted covariance matrix from the weighted samples  $\{(\bar{\omega}_j^{(k-1)}, \phi_j^{(k-1)})\}_{j=1}^N$ . As discussed by [88], the spread parameter is set as  $c_0 = 2.38/\sqrt{p}$ , since it allows for efficient transitions in Gaussian steps.

Once the first crumb is drawn, a first candidate  $\xi_1$  is generated from the appropriate Gaussian distribution

$$\xi_1 \sim \mathcal{N}(\varsigma_1, c_0^2 \Sigma_k), \quad (4.15)$$

where  $c_0$  is a spread parameter for the proposals and  $\Sigma_k$  defined as above. In general, the  $i$ -th candidate for the next state of the Markov chain can be generated as

$$\xi_i \sim \mathcal{N}\left(\bar{\varsigma}_i, \left(\frac{c_0}{i}\right)^2 \Sigma_k\right), \quad (4.16)$$

where  $\bar{\varsigma}_i$  is the average of the crumbs generated so far, as proposed by [165]. Note how the generation of new candidates in the slice is narrower as the candidates are rejected by means of the parameter  $c_0/i$ . However, the mean for the Gaussian proposal might not converge to a point in the slice if the posterior is a multi-modal distribution. To cope with this limitation, it is proposed to use a weighted average of the current state and the crumb centre to enhance the mixing of the sampler. Namely, by sampling the  $i$ -th candidate from a Gaussian distribution with mean

$$\bar{\varsigma}_i^* = \alpha_i \phi_0 + (1 - \alpha_i) \bar{\varsigma}_i \quad (4.17)$$

and covariance  $(c_0/i) \Sigma_k$ . The weight parameter  $\alpha_i$  can be defined in terms of the number of crumbs previously rejected. Since it is desirable that  $\alpha_i \rightarrow 1$  as  $i$  increases, we can define it either as  $\alpha_i = (1 - 1/i)$  or  $\alpha_i = (1 - \exp(-i))$ . As confirmed by these experiments,  $\alpha_i$  is linearly-dependent on the crumb iteration, since the exponential behaviour exhibits pronounced decay towards the current state, causing Random Walk behaviour. The sampling in each annealing level is summarised in Algorithm 4. To avoid cluttered notation, the conditioning on the design points  $\mathcal{D}$  is dropped in the remainder.

**Algorithm 4:** TA<sup>2</sup>S<sup>2</sup> at annealing level  $k$ **Input :**

- ◇  $\phi_1^{(k-1)}, \dots, \phi_N^{(k-1)} \sim p_{k-1}(\phi)$ , generated at previous level;
- ◇  $\phi_1^{(k)} \in \Phi$ , initial state of the chain;

**Output :**

- ◇  $\phi_1^{(k)}, \dots, \phi_N^{(k)} \sim p_k(\phi)$ ;

**begin**    Compute covariance matrix  $\Sigma_k$  from the weighted samples;    **for**  $i \leftarrow 1$  **to**  $N - 1$  **do**        Define slice  $S_{\phi_i}^k$  as in (4.10) ;        Define the crumb counter as:  $l \leftarrow 0$ ;        **do**             $l \leftarrow l + 1$ , and generate  $u \sim U(0, 1)$ ;            **if**  $|\mathcal{J}| \neq \emptyset$  **or**  $u < p_{\text{renew}}$  **then**                Choose random  $j$  from index set  $\mathcal{J}$ ;                 $\varsigma_l = \phi_j^{(k-1)}$  ;            **else**                Generate  $\varsigma_l \sim \mathcal{N}(\phi_i^{(k)}, c_0^2 \Sigma_k)$  ;            **end**            Define crumb as  $\bar{\varsigma}_l^* = \alpha_l \phi_i^{(k)} + (1 - \alpha_l) \bar{\varsigma}_l$  ;            Generate candidate  $\xi_i \sim \mathcal{N}(\bar{\varsigma}_l^*, (c_0/l)^2 \Sigma_k)$  ;        **while**  $\xi_i \notin S_{\phi_i}^k$ ;        Define new state of the chain  $\phi_{i+1}^{(k)} = \xi_i$ ;    **end****end****4.2.2 Overview of the Full Sampler - TA<sup>2</sup>S<sup>2</sup>**

The algorithm starts with a uniform sample in an admissible space  $\Phi$ , as implied by the *meta*-prior distribution in equation (4.8). As a second step, the algorithm described in the previous section is used to generate the samples of the first annealing level, that is  $\phi_1^{(1)}, \dots, \phi_N^{(1)} \sim p_1(\phi)$ . As mentioned before, this set of points allows one to approximate the slices in the next annealing level and the areas where the posterior mass is concentrated. As the sequence of temperatures converges to 1, it is expected that better approximations are generated. That is, until a sample  $\phi_1^{(k^*)}, \dots, \phi_N^{(k^*)}$  has been drawn and is uniformly distributed in the

set  $\Phi^*$ . The next section discusses how to learn the temperature sequence and the overall parallel implementation achieved by embedding it on a transitional Markov chain schedule.

### Annealing Schedule

The way the temperature sequence is determined is one of the most crucial aspects of any simulated-annealing-based method. It is clear that if the change of temperatures is abrupt the markers will degenerate quickly, as observed in sequential Monte Carlo samplers. On the contrary, if the sequence of temperatures decreases slowly the actual efficiency of the algorithm is hindered, since sampling in a sequence of annealing levels is redundant for the generation of posterior sample. Setting the temperature sequence beforehand requires prior knowledge of the overall behaviour of the function  $\mathcal{H}(\cdot)$  and the topology around the set  $\Phi^*$ , both of which are generally not available.

Following the suggestion by [226], the *Effective Sampling Size* can be used as a measure of degeneracy of the chain in each annealing level. This allows one to measure how similar the  $(k-1)$ -th and the  $k$ -th densities are. The effective sample size can be approximated by

$$\hat{n}_{\text{eff}} = \frac{1}{\sum_{i=1}^N \left( \bar{\omega}_j^{(k-1)} \right)^2}, \quad (4.18)$$

where  $\bar{\omega}_j^{(k-1)}$  is the normalised weight of sample  $\phi_j^{(k-1)}$ . Given the temperature of the previous level is known, the problem is to determine the temperature of the next one. This is done by determining a target threshold for  $\hat{n}_{\text{eff}}$  in terms of the size of the simulated set. Thus, given  $\gamma \in (0, 1)$ , the target threshold is defined by  $\gamma N = \hat{n}_{\text{eff}}$ . Rewriting this expression in terms of the unnormalised sample weights we obtain

$$\frac{\sum_{j=1}^N \exp \left\{ -2\mathcal{H} \left( \phi_j^{(k-1)} \right) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\}}{\left( \sum_{j=1}^N \exp \left\{ -\mathcal{H} \left( \phi_j^{(k-1)} \right) \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \right\} \right)^2} = \frac{1}{\gamma N}, \quad (4.19)$$

which yields an equation for the unknown temperature  $\tau_k$ . Solving the equation for  $\tau_k$  can be done efficiently by standard numerical techniques such as the bisection method.



The value of the threshold  $\gamma$  affects the overall efficiency of the annealing schedule. If a value close to zero is chosen, the resulting algorithm will create few tempered distributions and this will result in poor approximations. If  $\gamma$  is close to 1, then there will be excessive tempered distributions and redundant annealing levels. As suggested by [14], and as confirmed by the experiments carried out for this work, a value of  $\gamma = 0.5$  delivers acceptable efficiency.

### Parallel Markov Chains

As described so far, the proposed algorithm can be computationally expensive if the Markov chain of the samples is drawn sequentially. This is due to the inversion of a  $n \times n$  matrix and related products in equation (3.14). Hence, it is desirable to speed up the process of generating samples in each annealing level. In our context, the inversion of such a matrix is not prohibitive, since we assume that the set of training points is expensive to acquire, however, a fast sampling algorithm is desired for a complete Bayesian treatment of the problem. This way we can compensate for the drawbacks associated with an appropriate error estimation by using the emulator in a Bayesian setting [see 130, for a discussion]. The idea of parallelisation comes from an adaptation of Transitional Markov Chain Monte Carlo (TMCMC) [45] in the context of the annealed adaptive slice sampling algorithm described previously.

TMCMC builds a Markov chain from a target distribution in a sequential schedule as in Sequential Monte Carlo [55, 104] or Particle Filtering [6]. That means that  $N$  Markov chains are started, each from the state of an initial Markov chain being drawn from the prior distribution of the Bayesian inference problem. The key difference is that the Markov chains are allowed to communicate among each other by a *transition* mechanism that allows growing each Markov chain differently within the same annealing level, disregarding poor initial states for certain Markov chains. The length of the Markov chain is determined by a probability proportional to the importance sampling weight defined in equation (4.11). By doing so, the markers are automatically selected in the updating sequence and concentrated around the modes found during the annealing. This improves the mixing of the samples generated in each annealing level.

Summarising, the proposed TA<sup>2</sup>S<sup>2</sup> algorithm consists of Markov chains generated as established in Algorithm 4, the annealing temperature being determined

empirically by the effective sampling size described in Section 4.2.2 and stopped whenever the temperature reaches 1. The selection of the initial states of the Markov chains and their growth length is a direct implementation of the TMCMC method [45] for Bayesian model updating.

### 4.3 Numerical Experiments using $TA^2S^2$

The following examples illustrate the effectiveness and robustness of  $TA^2S^2$  when sampling the hyperparameters of Gaussian process emulators. The first example is Franke's function [107], which can have challenging features when emulated. The second example is a five-dimensional model [168] which has been previously used to test Gaussian processes meta-models [13]. The third example is a ten-dimensional model for the weight of a wing of a light aircraft [80]. Concerning the nugget of the surrogate, sigmoid transformation is performed in order to sample all covariance hyperparameters with multivariate Gaussian distributions as discussed in Section 4.2.1. That is, auxiliary component  $z_\delta$  is introduced and extend the vector of hyperparameters  $\phi$  to  $\mathbb{R}^{p+1}$ . That is,  $z_\delta$  is the  $(p+1)$ -th component to be sampled in the algorithm. Finally, the nugget is transformed back to the original scale as

$$\theta_\delta = \frac{1 - l_b}{1 + \exp(-z_\delta)} + l_b, \quad (4.20)$$

where  $l_b$  is the lower bound, which is set equal to  $10^{-12}$  following the discussions in [185]. To incorporate the algorithm to the length-scale hyperparameters, the sampling has been performed in logarithmic space to avoid additional concerns for the non-negative restrictions imposed to the aforementioned variables as in other sampling schedules [162]. The initial values of the algorithm, equation (4.8), are set to a uniform distribution in a wide practical range, that is the interval  $[-7, 7]$  for the length-scales. For the nugget, a non-informative truncated beta distribution in the interval  $[l_b, 1]$  has been considered.

The code was implemented in MATLAB and all examples were run in a GNU/Linux machine with an Intel i5 processor with 8 Gb of RAM. For the purpose of reproducibility, the code used to generate the examples in this thesis is available for download at [http://github.com/agarbuno/ta2s2\\_codes](http://github.com/agarbuno/ta2s2_codes).

In order to contrast the proposed methodology with existing ones, the Particle Learning sampler PLGP [103] is taken as as a benchmark. This sampler has

proven effective for sampling the posterior distribution of the hyperparameters of a Gaussian process by means of tempering in a data-oriented manner, *i.e.* by feeding subsets of the training runs in each annealing level. Although the proposed sampler can be implemented in an on-line fashion akin to PLGP, we resort only at comparing them as strategies in batch applications. Note that the extension to on-line learning tasks can be done by regarding the posterior of a subset of data as the prior for the next set of training runs. This can be followed easily as the re-weighting of the samples by a data-oriented alternative to equation (4.11) can help adjust the importance of the samples.

Following the discussion of [133] proper scoring rules should be used in order to compare the probability statements made by the Gaussian process model resulting from the samples used to marginalise the predictive posterior. In the context of Gaussian processes, both prediction and error estimation are used to assess the quality of the surrogate, *i.e.* the estimated mean and variance. If a local scoring rule such as the negative logarithm of predictive density (NLPD) is used to evaluate the generated samples, there is a risk of penalising heavily over-confident predictions and treat with less rigour under-confident far-off predictions. This is not desirable since it is known that the full Bayesian treatment in Gaussian processes is preferred for better error estimation in uncertainty analysis [130]. In contrast, by using distance-sensitive scoring rules such as the continuously ranked probability score (CRPS) there is better placement of probability mass near target values, although not exactly placed at the target. It is defined as

$$\text{CRPS}(F, x) = \int_{-\infty}^{\infty} (F(y) - \mathbf{1}\{y \geq x\})^2 dy, \quad (4.21)$$

where  $F$  is the cumulative predictive distribution and  $x$  is the point where it is verified. A Gaussian approximation is assumed for the predictions made by the Gaussian process emulator. This uses the mixture model expressed in equations (3.10) and (3.12) to be able to use the complete mixture expression developed in [105] which is included for completeness. That is, for a mixture of Gaussians the CRPS can be written as

$$\begin{aligned} \text{CRPS} \left( \sum_{m=1}^N \omega_m \mathcal{N}(\mu_m, s_m^2), x \right) &= \sum_{m=1}^N \omega_m A(x - \mu_m, s_m^2) - \\ &\quad \frac{1}{2} \sum_{m=1}^N \sum_{n=1}^N \omega_m \omega_n A(\mu_m - \mu_n, s_m^2 + s_n^2), \end{aligned} \quad (4.22)$$

where  $\omega_i$  denotes the weight of the sample,  $\mu_i$  is the mean of  $x$  given by sample  $i$  and  $s_i^2$  the corresponding estimated variance. The function  $A(\cdot, \cdot)$  is defined as

$$A(\mu, \sigma^2) = 2\sigma f_{\mathcal{N}}\left(\frac{\mu}{\sigma}\right) + \mu \left(2F_{\mathcal{N}}\left(\frac{\mu}{\sigma}\right) - 1\right), \quad (4.23)$$

where  $f_{\mathcal{N}}(\cdot)$  and  $F_{\mathcal{N}}(\cdot)$  denote the density and cumulative functions of a standard Gaussian random variable. It should be noted that the CRPS does not possess an analytic expression for every probability function used for prediction [105], thus the choice of using a mixture of Gaussians for the predictive posterior distribution instead of  $t$ -distributions.

Other alternatives for scoring rules for Gaussian processes could be the bootstrapped variance predictor of [59]. This predictor aims to estimate the variance of the Gaussian process independent of the set of points used for training. However, such approach leads to prefer under-confident predictions not exactly around the target value and relies in the assumption of infinite repeatability of the simulator experiments. This hypothesis is not satisfied by Bayesian analysis of computer code output (BACCO), since by assumption, the generation of training runs is limited due to computational cost.

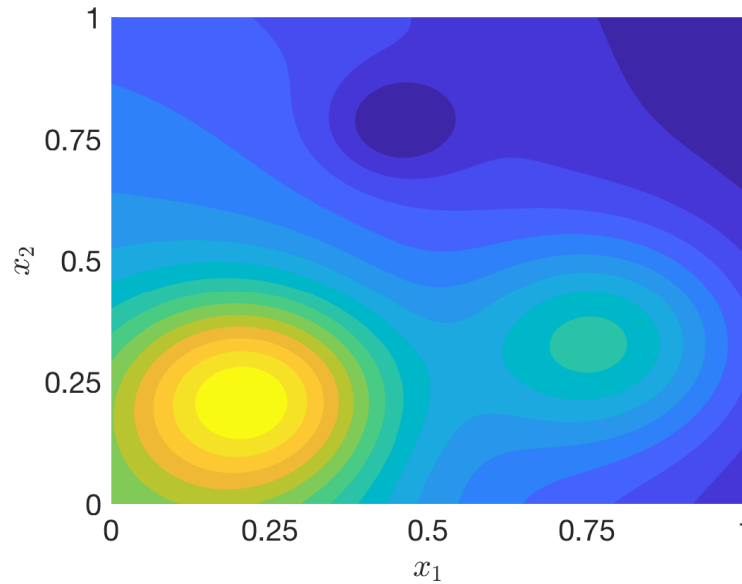
### 4.3.1 Franke's function

Franke's function has been used to test Gaussian process emulators [107]. Its complexity stems from the presence of two peaks and one dip in its landscape. Let  $f : [0, 1]^2 \rightarrow \mathbb{R}$  be such that

$$\begin{aligned} f(\mathbf{x}) &= 0.75 \exp\left(-\frac{(9x_1 - 2)^2}{4} - \frac{(9x_2 - 2)^2}{4}\right) + 0.75 \exp\left(-\frac{(9x_1 + 1)^2}{49} - \frac{9x_2 + 1}{10}\right) \\ &\quad + 0.5 \exp\left(-\frac{(9x_1 - 7)^2}{4} - \frac{(9x_2 - 3)^2}{4}\right) - 0.2 \exp(-(9x_1 - 4)^2 - (9x_2 - 7)^2). \end{aligned} \quad (4.24)$$

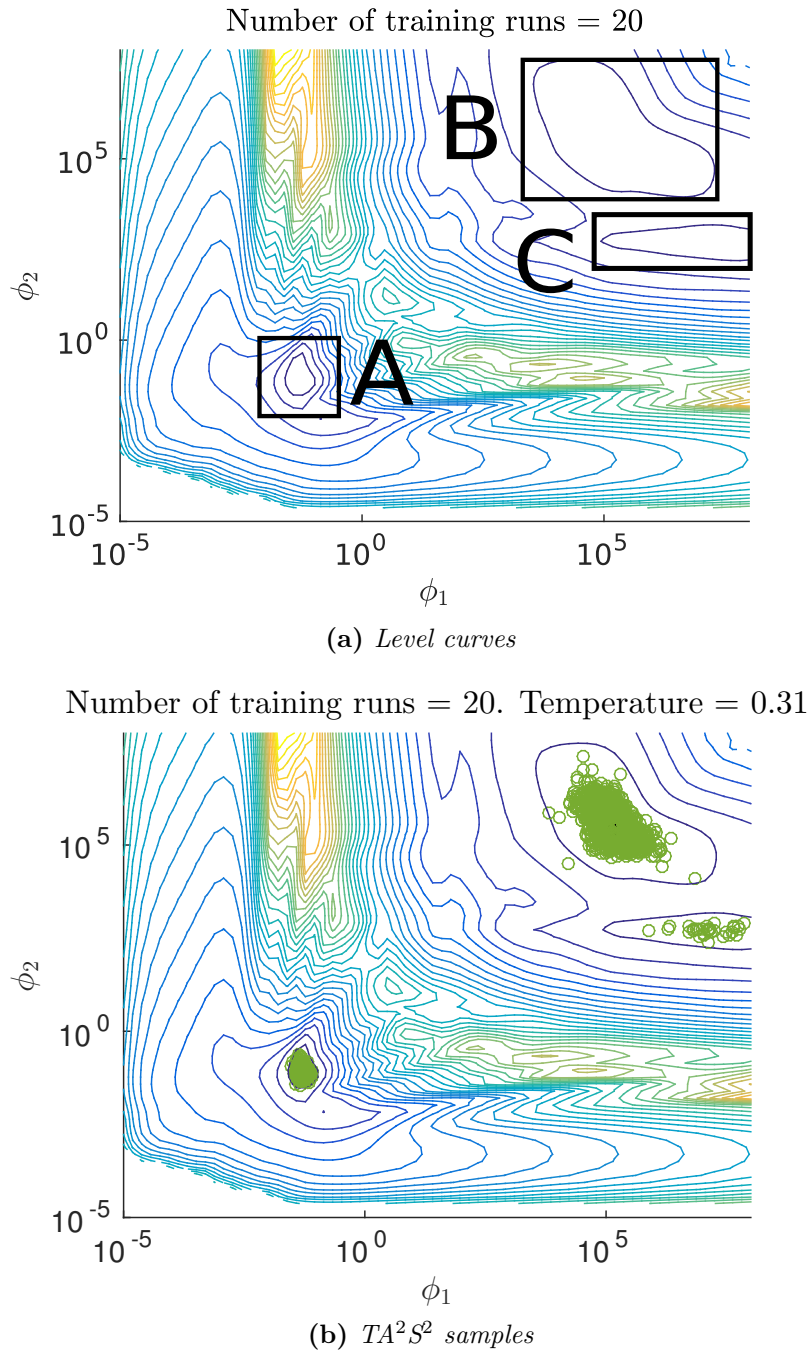
For completeness, Figure 4.1 depicts the contour levels of Franke's function in the region of interest. In order to train the model 20 design points were chosen

using a Latin hypercube sampling (LHS) scheme. For testing purposes, 100 independent design points were chosen by an independent LHS. Figure 4.2a shows the multi-modal integrated log-posterior for a fully-parametrised Gaussian process [85]. Region A contains a mode with no preference for any dimension. Regions B and C depict different asymptotic behaviours of the emulator. In region B, the emulator behaves as linear regression model, as noted by [2]. Region C corresponds to a model which disregards the first dimension. In Figure 4.2b a set of samples obtained by applying  $TA^2S^2$  is showed, illustrating the ability to overcome possible multi-modal distributions that arise in Bayesian analysis of expensive computer codes.



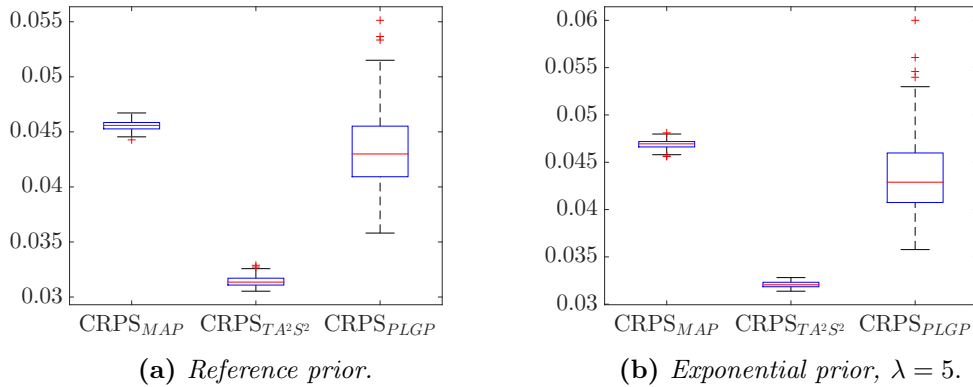
**Figure 4.1:** *Contour levels of Franke's function*

To contrast the proposed sampler against the PLGP benchmark, a set of 100 experiments were run. In each experiment, a sample of size 100 of length-scale hyperparameters was obtained by each method. This sample size was achieved by thinning the  $TA^2S^2$  results when a chain of length 2000 was constructed in every annealing level. The quality of the probability statements made from both results were compared by means of the CRPS, as depicted in Figure 4.3. Note that the training set was the same for each experiment and variations in the results among experiments are mainly because of the stochastic nature of the sampling schemes. In Figure 4.3a the boxplots for the CRPS computed from the



**Figure 4.2:** Projection of the negative log-posterior curves in the two dimensional length-scale space for Franke's simulator using a fully-parametrised Gaussian process. The minimum possible value of  $10^{-12}$  for the nugget  $\phi_\delta$  has been used for the projection. Depicted in Figure 4.2b, the temperature has been decreased beyond 1 to retrieve samples from the posterior modes.

samples are shown. Figure 4.3b shows the same plots but with an exponential prior for all hyperparameters with rate 0.2, *i.e.*  $\lambda = 5$ . This choice of a prior distribution was made since the PLGP software assumes an exponential prior for both the length-scales and nugget term [103]. In both settings, the proposed sampler outperforms both the PLGP alternative and the MAP estimate. The latter was calculated from the samples generated by  $TA^2S^2$ . These experiments demonstrated that the MAP estimated this way usually corresponds to the one found by local optimisation routines such as Nelder-Meade or BFGS. The variation in the scores of the MAP illustrates the multi-modality properties of the integrated posterior. All the MAP estimates reported in the remainder are calculated based on this observation. Additionally, it can be seen that the PLGP results seem to contain those achieved by the most probable candidate. In this experiment either using a sample from PLGP or MAP translate in comparable results.



**Figure 4.3:** Boxplots of CRPS comparing MAP,  $TA^2S^2$  and PLGP. In both cases the proposed sampler outperforms PLGP.

### 4.3.2 Nilson-Kuusk model

The simulator for reflectance of a homogeneous plant canopy has already been introduced in Chapter 3. In this experiment, both samplers were used to train a Gaussian process emulator with a dataset of 100 simulation runs. The test set consisted of a different set of 150 training runs. Both datasets, whose design points were generated through LHS, were obtained from the GEM-SA software web page (<http://ctcd.group.shef.ac.uk/gem.html>). On average, a total of

10 tempered distributions were used in the annealing schedule, while keeping the sampling as  $N = 5000$  in each level. A thinned sample of 100 experiments was recovered by the end of each  $TA^2S^2$  run to compare results.

The results shown in Figure 4.4 demonstrate again the overall improved performance of using the proposed sampler in contrast with the benchmark. In this case, one set of experiments (50 iterations) consisted on making inference with the reference prior discussed previously, while the second set (50 iterations) used a common exponential prior. Both samplers outperform the MAP estimate which provides evidence that a more complete uncertainty analysis can be carried out if instead one turns to a full Bayesian inference scheme. Additionally, it is worth noting that Figures 4.4a and 4.4b show that the MAP changes with the prior used. This is not a matter of concern, as it is consistent with the notion that small amount of data is being used for inference and the probabilistic model of the observables is not dominating the prior beliefs.

### 4.3.3 Wing weight model

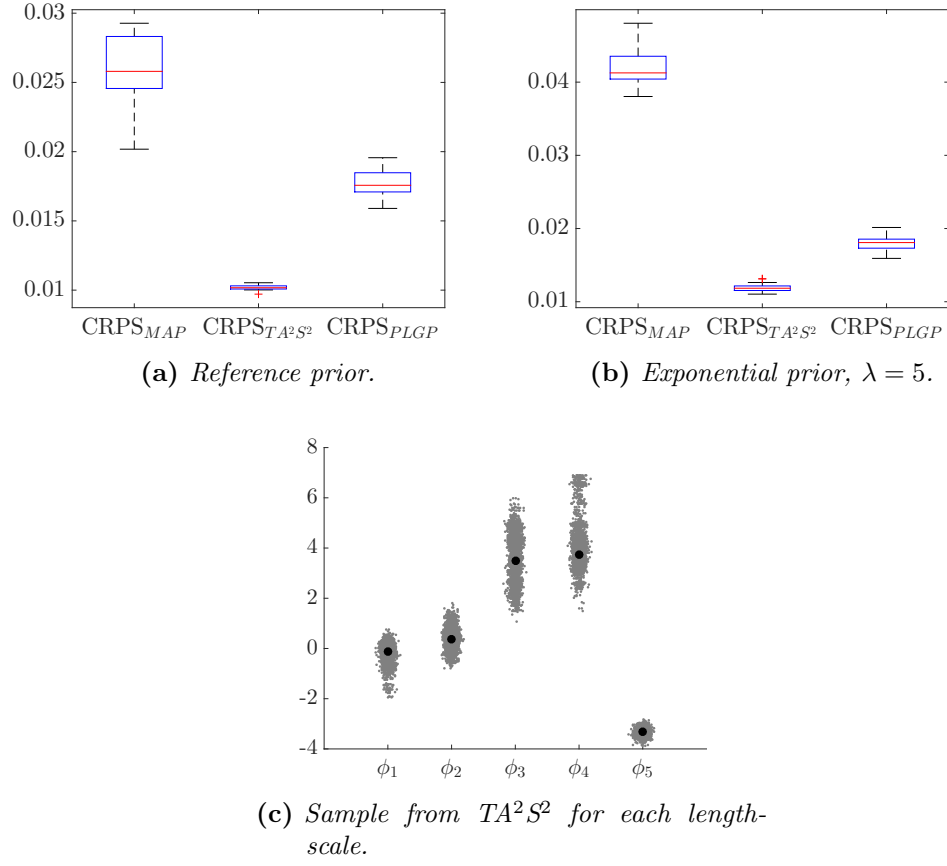
This simulator of the weight of the wing of a light aircraft [80] has been used for input screening, that is to identify the most important input parameters affecting the output. Coupled with a Gaussian process emulator with the squared exponential kernel in equation (3.3), a sensitivity analysis of the wing weight with respect to each input variable can be performed. The model is given by

$$f(\mathbf{x}) = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left( \frac{A}{\cos^2(\Lambda)} \right)^{0.6} q^{0.006} \lambda^{0.04} \left( \frac{100 t_c}{\cos(\Lambda)} \right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p, \quad (4.25)$$

where the input variables and the range of their values are summarised in Table 4.1. For this problem, the evaluation of the reference prior is prohibitive since it scales with the number of dimensions [179]. Thus, a uniform prior in the hyperparameters' log-space has instead been used for this experiment.

The inputs were rescaled to the 10-dimensional unit hypercube  $[0, 1]^{10}$ . Two LHS samples of size 100 and 300 were chosen as training and testing sets respectively. At each annealing level, 5000 samples were generated, achieving convergence after 15 levels on average. As before, a thinned sample of 100 was kept to compare results with the benchmark in each experiment. A total of 50 experiments we run





**Figure 4.4:** Nilson-Kuusk results obtained after 50 experiments were run. The scatter plot (4.4c) is drawn from a single experiment. The MAP estimates are shown in black.

Input	Range	Description
$S_w$	[150, 200]	Wing area
$W_{fw}$	[220, 300]	Weight of fuel in the wing
$A$	[6, 10]	Aspect ratio
$\Lambda$	[-10, 10]	Quarter-chord sweep
$q$	[16, 45]	Dynamic pressure at cruise
$\lambda$	[0.5, 1]	Taper ratio
$t_c$	[0.08, 0.18]	Aerofoil thickness to chord ratio
$N_z$	[2.5, 6]	Ultimate load factor
$W_{dg}$	[1700, 2500]	Flight design gross weight
$W_p$	[0.025, 0.08]	Paint weight

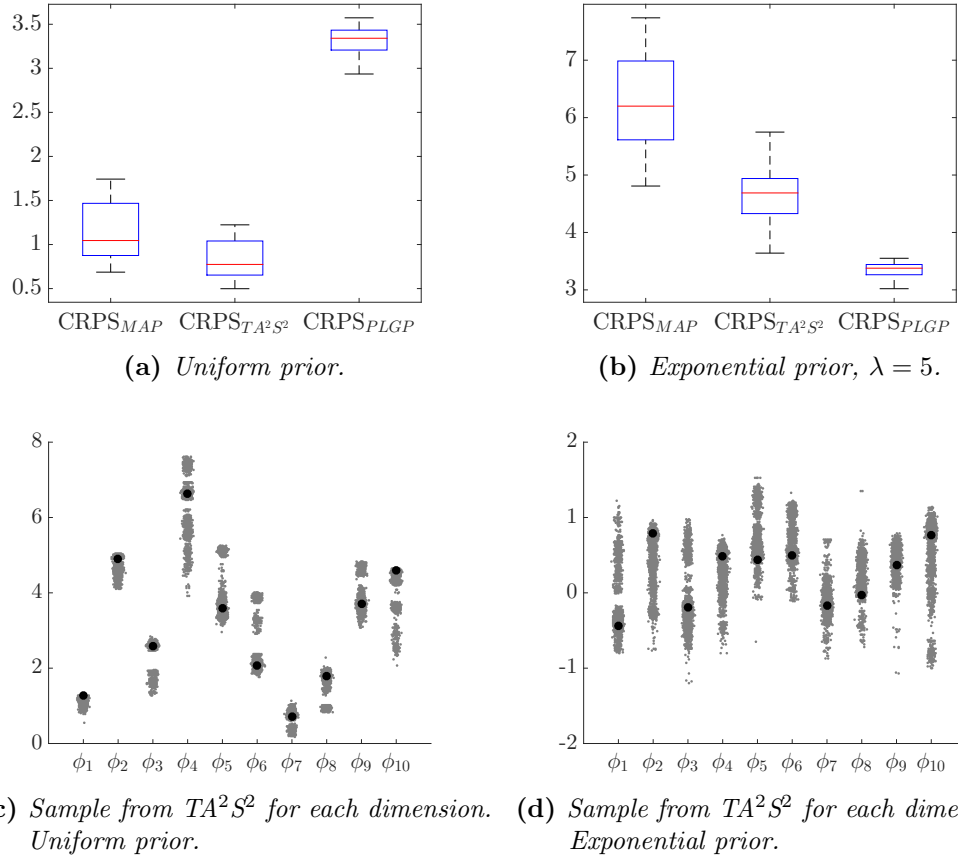
**Table 4.1:** Inputs of the wing weight model.

in this case.

In Figures 4.5a and 4.5b, it can be noted that by using sampling one can obtain an improved version of the probabilistic statements made by the surrogate. Figure 4.5a shows that the proposed sampler outperforms the benchmark, concentrating its samples around the mode of the posterior distribution. This is shown by the location and spread of the CRPS for both  $TA^2S^2$  and MAP results, which are similar. Although the CRPS with  $TA^2S^2$  exhibits better performance as can be seen from the location and spread of the boxplot. PLGP in the case of Uniform priors seem to be sampling from other areas with less spread than that of the proposed algorithm. However, if an exponential prior distribution is used for both length-scales and nugget, as in Figure 4.5b,  $TA^2S^2$  is outperformed by the benchmark. Note how the location of the mode changes dramatically, showed by both the means of the boxplot of the CRPS of the MAP, and by the difference of the samples plotted in Figures 4.5c and 4.5d. Nonetheless, the proposed sampler is capable of offering improved probabilistic statements for the regression task compared to the MAP estimate.

## 4.4 Discussion on $TA^2S^2$ Results

The Transitional Annealed Adaptive Slice Sampling ( $TA^2S^2$ ), has been introduced to sample from the posterior distribution of the hyperparameters of a Gaussian process. In this context, it is known that multi-modal distributions are encountered.  $TA^2S^2$  combines Slice Sampling for delayed-rejection, Transitional Markov Chain Monte Carlo (TMCMC) for efficient parallelisation and Markov chain growth, and Asymptotically Independent Markov Sampling (AIMS) for driving the annealing schedule. The delayed-rejection feature of Slice Sampling provides improved mixing which is desirable in highly correlated spaces. Additionally, the proposed algorithm provides a sampling scheme with no burn-in periods for local Markov chains. This is computationally advantageous in Gaussian process applications where the computational burden of the sampling scheme is dominated by the inversion of the correlation matrix. However, it should be noted, that there is an implicit burn-in period associated with high temperature levels. Nonetheless, efficient coverage of the sampling space is achieved by the annealing schedule and the incorporation of Slice Sampling in Sequential Monte Carlo (SMC) methods



**Figure 4.5:** Results for the wing-weight model using the  $TA^2S^2$  and PLGP algorithms. The scatter plot (4.4c) is drawn from a single experiment, in black the MAP estimates.

[55]. Simulated tempering [149] and tempered transitions [161] are two methods closely related to simulated annealing, although they are based on importance sampling principles. Both algorithms provide an alternative annealing schedule to AIMS and are characterised by treating the temperature parameter as a random variable. In particular, this type of algorithms allows one to explore different annealing levels within one Markov chain. This is different to the algorithms proposed in Chapters 3 and 4, as PAIMS and  $TA^2S^2$  explore sequentially the annealing levels and do not reintroduce previously explored temperature values. This provides an alternative direction of future research.

The examples presented show how  $TA^2S^2$  is capable of efficiently exploring multi-modal distributions providing a better alternative to MAP estimates or traditional MCMC methods for Gaussian processes applications. Efficiency is gained through the adaptive nature of the algorithm which allows to sample from complicated regions of the posterior distribution. Namely, modes separated by large valleys of low probability. Furthermore,  $TA^2S^2$  allows the generation of samples more efficiently than in traditional MCMC applications, where longer chains are needed to ensure good coverage of the sampling space. This is particularly relevant in the context of computer experiments and Engineering, where it is usually considered that the cost of marginalising the hyperparameters does not justify the additional computational burden. The proposed method justifies the cost of sampling the hyperparameters by providing robust estimates of the predicted error. This is reflected through the continuously ranked probability score (CRPS) which shows that the marginalised emulator outperforms MAP estimates. Moreover, the proposed sampling scheme performs as good as the PLGP benchmark, which justifies the annealing by means of functional tempering rather than by subsets of data, which is also sensitive to data permutations.

The proposed algorithm could also be employed for global optimisation problems, by allowing the temperature to reach a practical zero. This has application in other areas of Bayesian inference problems and machine learning. As discussed previously, the algorithm can be used in active learning schedules by means of Bayes' Theorem.

The computational cost of the proposed sampler is dominated by the inversion of the covariance matrix in the integrated posterior distribution. This limitation is inherent to any MCMC schedule used for Gaussian processes. Further strategies to improve the speed of the sampler could be developed, such as the exact evaluation

of the log-posterior only for candidates where the probability of acceptance is high. For such strategy, a first order Taylor expansion of the objective function could be a feasible enhancement. This is material for future research.

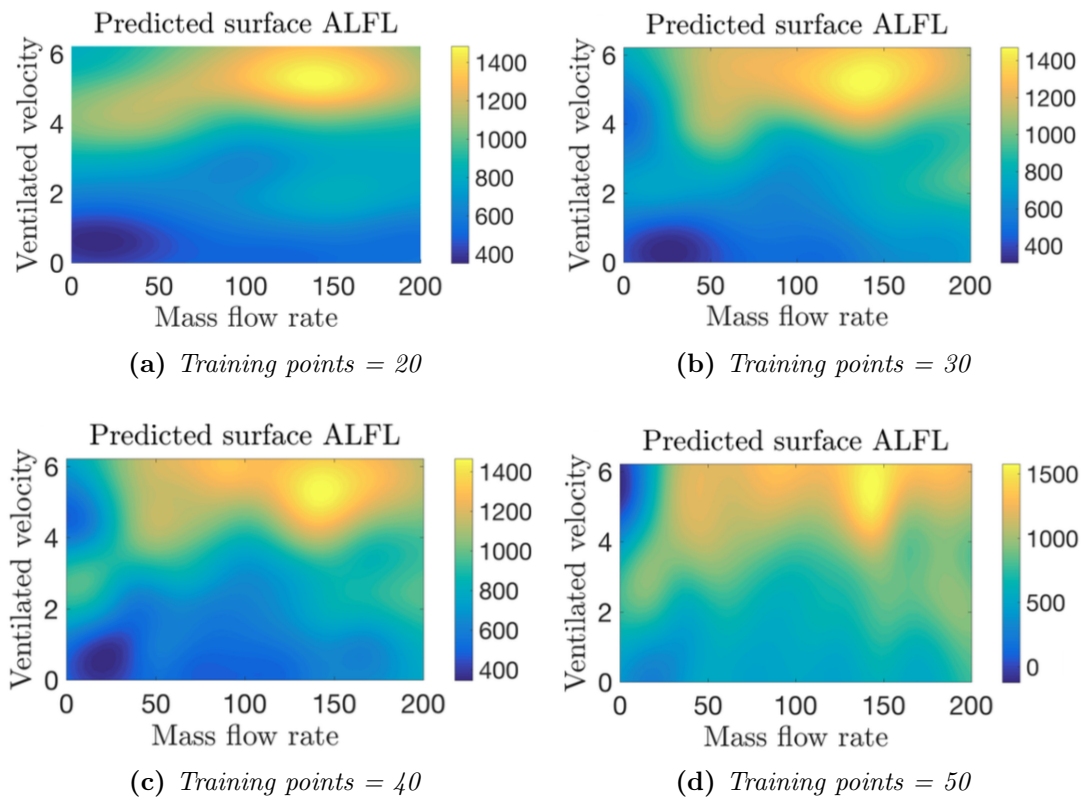
## 4.5 Industrial application

This section illustrates the use of a full Bayesian Gaussian process as an emulator of a computationally expensive simulator model. This work was done with MMI Engineering as part of a industrial partner collaboration at the Institute for Risk and Uncertainty at the University of Liverpool. The aim of this exercise was to build a fast surrogate model for a Computational Fluid Dynamic (CFD) simulation of a gas leakage in an offshore platform designed for explosion risk analysis. The objective was to enable fast exploration of the input configuration space of the simulator in order to refine their understanding of the CFD model under consideration.

The simulator was originally a highly multi-dimensional model in both inputs and outputs. The input parameter space consisted in wind directions, gas flow rate, wind velocity, release location, and direction of release. The output consisted of the gas density contained at every grid cell in the platform for the duration of the simulation. For simplicity, and after careful elicitation with the MMI team, the input space was reduced to a 2-dimensional simulator with mass flow rate and ventilated velocity as input variables of interest. The considered input domain was defined as the rectangle  $[0, 200] \times [0, 6.22]$ , as shown in the axis labels of Figure 4.6. The direction of release was determined as against a wall in the structure to allow a greater concentration of gas with collinear wind direction. The chosen output of interest was the maximum size of a gas cloud formed above the lower flammable limit (ALFL). In order to build an emulator for the Gas dispersion model, up to 50 possible configurations were chosen following a Sobol sequence of random numbers[201]. This design offered the advantage of being a nested sequence of numbers such the first  $n$  points were contained within the next  $n + r$  points, and having space-filling properties. This provided a testing environment for the GP emulator.

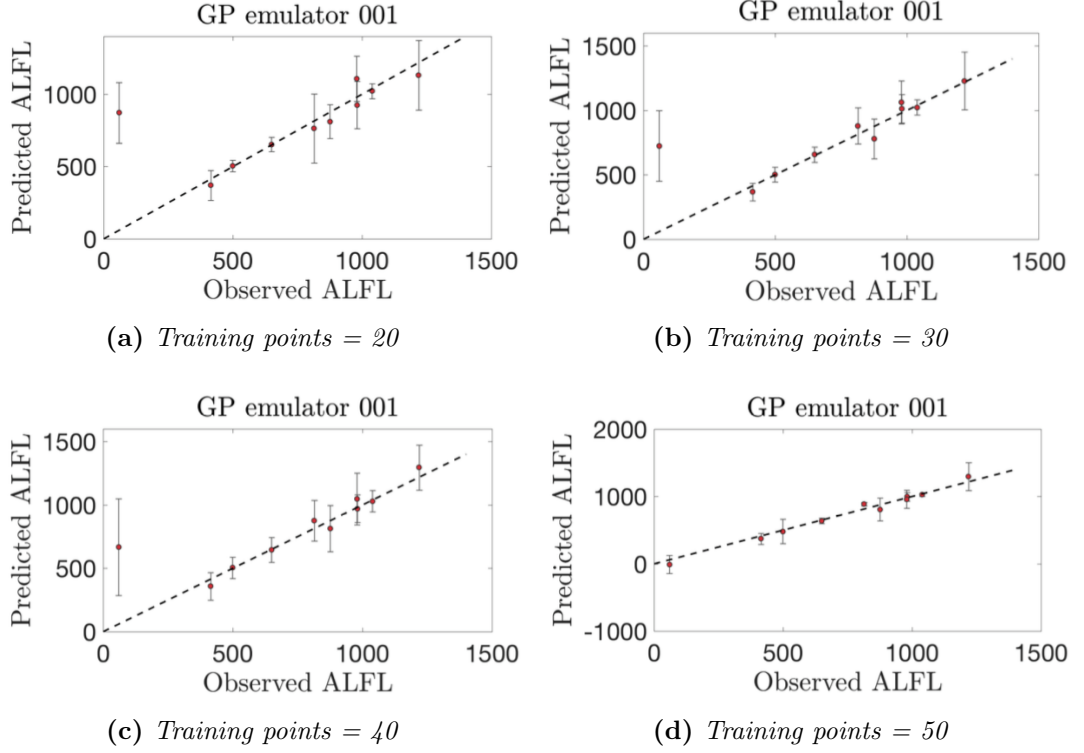
At an initial stage, 20 points were used to train a GP emulator and 10 additional points were chosen as a validation set (the validation set was selected independently

from the training set). A zero mean GP was used with covariance function chosen from the Matérn family with smoothness degree defined as  $5/2$ . This assumed a moderate degree of smoothness in the output of the simulator with respect to the inputs locations. This means that even though two configurations are close to each other in input space, the output of the simulator might not be as smooth due to the nature of the numerical methods that conform the gas dispersion simulator. This was often seen as one of a pair of nearby input locations resulted in a non-converging simulation within the allotted time. Each CFD simulator run was given 2 hours of running time. The GP was trained by a fully Bayesian strategy. This strategy provided a principled estimation of the uncertainty of fitting an emulator with such limited data and time constraints. The results shown in Figure 4.6 demonstrate the ability of the GP emulator to create a surface that interpolates the information provided by the training runs.



**Figure 4.6:** Prediction of the the maximum gas cloud being formed in the platform for any configuration in the space of interest with increasing number of training runs.

Moreover, a region of mismatch between emulation responses and simulator output was identified due to a low mass being dissipated by rapid ventilation as shown in Figure 4.7. In order to assess the improvement of the GP as more training points were added, the hierarchical structure of the Sobol sequence was exploited adding 10 points at a time from the remainder of the large Sobol design. It should be noted how even after adding more training runs the GP emulator was not able to capture the true predicted value for low mass flow rate and high ventilated velocity. Nonetheless, wide uncertainty bounds were still present in the middle stages of the emulation procedure.



**Figure 4.7:** Validation check from the predictions at different levels of training points. As the number of points is increased to train the emulator it's ability to predict the hard case is improved. Additionally, the uncertainty estimated in other regions of the input space is reduced while increasing the uncertainty in difficult areas.

This resulted in a valuable output for the MMI Engineering team, as their initial assumptions of the behaviour of the simulator were not accurate. In particular, this emulation exercise provided greater insight of the simulator responses near

the boundaries of the input space. This is because the output from gas dispersion with small masses and high ventilated velocities was not previously well studied for their in-house solver. Thus, this emulation exercise suggested further inspection of the simulator in such scenarios. In the end, the MMI team was able to understand better the dynamics of the gas cloud formation in the presence of limited computational resources.

## 4.6 Conclusion

In this chapter a new algorithm was proposed to sample from the posterior of the hyperparameter of a Gaussian process emulator. The ability to sample from multi-modal posterior distributions is due to the foundation in simulated annealing. Furthermore, the sampling method is flexible enough to allow one to target optimisation tasks by following an analogue of subset optimisation. This flexibility is exploited in history matching applications. First, a sampling version of the  $TA^2S^2$  algorithm is used for the emulation of computationally expensive computer codes. Second,  $TA^2S^2$  is used in an optimisation setting to find input regions of interest in the history matching formulation. This is presented and implemented in the context of history matching in [Chapter 5](#).



## CHAPTER 5

---

### Probabilistic Extensions to History Matching<sup>1</sup>

---

Scientific understanding of real world processes has dramatically improved over the years through computer simulations. These computational codes are parametrised by a set of values which are essentially a vector of input values. The previous chapters focused on tackling the problem of efficient exploration of these simulators. However, no matter how sophisticated or efficient, a simulator needs to be well-calibrated to experimental data if it is to be trusted. Thus, the validity of using a particular simulator to draw accurate conclusions, relies on the assumption that the computer code has been correctly calibrated. That is, the vector of input parameters is well known, and there is confidence that these values are able to replicate the process the simulator is modelling. This calibration procedure is often pursued under extensive code experimentation and it can be coupled with expert domain knowledge.

For some models the collection of data is so expensive that only a handful of experiments is feasible. This is common in applications such as astrophysics [217], epidemiology [3], and climate modelling [191], to name a few. History matching is a calibration technique that iteratively discards regions of the input parameter space through the use of an implausibility measure. This way, history

---

<sup>1</sup>The results and ideas developed in this chapter have been submitted for publication as a manuscript, see Garbuno-Inigo et al. [86]

matching overcomes the limited availability of data from the experimental process and is able to identify regions of parameter space that are likely to replicate the observed phenomena. Thus, the regions that show a high implausibility measure can be discarded from the analysis. The history matching procedure is also able to determine if there is no such region of interest. This can be used as evidence that further improvement in the simulator is needed. This contrasts with typical Bayesian analysis of computer code output (BACCO, [130]) where positive posterior probability mass is put in regions that under the history matching framework would otherwise be discarded.

As the mathematical models have increased in complexity, their associated simulators have become computationally expensive as well. For example, the simulators used for climate models, nuclear reactor models, and biological models complete a single simulation in a days [199]. This represents an additional layer of complexity as the ability to explore the input parameter space is dampened by the high computational costs of running the simulator. Common simulation techniques such as the Monte Carlo method and its variants are not well suited in this context. In turn, fast but accurate surrogate models of the simulator are needed to overcome this limitation. In particular, Gaussian processes (GPs) have been successfully used as surrogates in different scientific applications such as machine learning, spatial data analysis, genetics, and stochastic finite element models [186, 51, 124, 64], to name a few.

The use of both computer codes as simulators and surrogate models to approximate them, introduce a wide range of uncertainties in the modelling process. In history matching these uncertainties are elicited and incorporated in the variability of the predicted surrogate output. As such, the implausibility function is a measure of the number of standard deviations between the observed data and the expected output from the surrogate. In the literature, it has been common to use pointwise estimates of the implausibility function. For example, this is achieved by using the mean and variance estimators of a GP surrogate in pointwise evaluations of the implausibility function. To the author's knowledge, the full probabilistic characterisation of the implausibility as a random outcome from the GP model has not been widely explored in history matching applications.

In contrast, in the area of robust optimisation of black box computer codes, the probabilistic output of the GP is acknowledged and incorporated in the exploration of the input parameter space. For example, the optimisation is performed in

---

an iterative procedure that uses a GP model as a surrogate for the black box function [123]. In these applications, acquisition function with a probabilistic characterisation are used to guide the exploration of the parameter space. In particular, the expected improvement is guaranteed to find the optima of a function under certain regularity conditions [215]. As a consequence, the expected improvement criteria is often preferred to pointwise estimates of improvement to guide the exploration [80]. Motivated by this analogy, the history matching strategy developed in this chapter uses a full probabilistic characterisation of the output of the simulator to be able to guide the reduction of the input space.

History matching is a sequential procedure that identifies regions of nonimplausible input configurations to be able to refocus the surrogate model. This refocusing strategy enables a better identification of nonimplausible points by using an improved surrogate in the neighbourhoods of interest. This raises the question of how to choose new simulator runs to improve the current surrogate. In this chapter, the use of certain functions to guide the selection of points is proposed. These functions are known in other research communities as active learning criteria [74], sampling criteria [19] and learning functions [145]. In this work, the term active learning is used, as such functions are used to generate educated guesses on where to focus the limited computational resources for the matching problem.

This chapter proposes the use of a full probabilistic characterisation of the implausibility measure to refocus the surrogate of the simulator. Additionally, three active learning criteria are generalised and presented to guide the selection of new training runs to improve the surrogate model. First, the expected contour improvement of Ranjan et al. [184], which was specially designed to refine a surrogate model for a given contour level. Second, the expected risk of Echard et al. [73] used for reliability analysis and modified here to adapt to more a general contour estimation. Third, the entropic profile presented by Lv et al. [145] which is also modified to target a specific contour level of the surrogate model.

The chapter is organised as follows. In Section 5.1 the preliminaries for history matching are presented, with a brief overview of Gaussian processes as surrogate models. In Section 5.2 the identification of nonimplausible regions is discussed within the context of the simulated annealing sampling methods used in subset optimisation. This provides regions of input parameter space that are likely to match the simulator output to observed data. In Section 5.3 the three active

learning criteria used are presented. In Section 5.4 some illustrative examples are shown. Concluding remarks are presented in Section 5.5.

## 5.1 History matching

History matching is a calibration technique particularly useful in settings where not only the simulator model is computationally expensive, but the data-generating process is expensive as well. The seminal papers [48] and [49] introduced history matching as a pre-calibration technique used within the framework of Bayes linear to analyse computationally expensive computer models. Vernon et al. [216] presented a thorough exposition of history matching in large-scale high-dimensional applications such as the ones encountered in astrophysics. In the latter application, history matching was able to identify regions of input parameter space where a high-dimensional simulator output matches available data.

In particular, this dissertation will study the use of history matching in cases where the cost of generating new experimental data is so high that very limited information from the physical process under study can be recorded. In this setting, history matching aims to identify regions of the input parameter space  $\mathcal{X}$  of the simulator that are able to replicate the measured data. As mentioned in Chapter 1, this corresponds to a relaxation of the search for  $\mathbf{x}^*$  as the single optimal point that allows one to match the simulator output to the real physical process.

Let  $y$  denote the true physical process of interest, although only partially observed. For example, because of corruption in the recording mechanism. Let  $z$  denote the noise corrupted version of the process. That is,  $z = y + \epsilon_{\text{me}}$ , where  $\epsilon_{\text{me}}$  denotes an observational noise with zero mean and finite variance. The limitation of not being able to observe directly the data is what is commonly referred to as *observation uncertainty* or *measurement error*. The quantity of interest,  $y$ , is assumed to be a functional output of the simulator being calibrated. Let  $\eta(\mathbf{x})$  denote the simulator output using the input parameter  $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$ . The simulator  $\eta(\cdot)$  is assumed to be only a mathematical abstraction of the true underlying process which adds an additional layer of uncertainty in the computer output. The mismatch between the computer model and the process is termed as *model discrepancy* as in [130]. Let  $\epsilon_{\text{md}}$  denote the model discrepancy and assume  $y = \eta(\mathbf{x}^*) + \epsilon_{\text{md}}$ . Note that the model discrepancy is usually modelled after

correctly calibrating the simulator. That is, the model discrepancy is usually modelled after identifying input  $\mathbf{x}^*$ , the optimal calibration point. The source of such discrepancy could be a consequence of random numbers used in the simulator, numerical accuracy of floating point arithmetic, oversimplified assumptions of the computational model, among others. Therefore, it is usually elicited from domain expert knowledge [4].

The computational complexity of the simulator inhibits our ability to explore the configuration space. In typical industrial applications each run of the simulator could take as much as days or weeks to complete. As a consequence, an additional layer of uncertainty is introduced. In the literature this is known as *code uncertainty*. An inexpensive approximation for the simulator is used to cope with this limitation. In this work, the surrogate used for the simulator is a full Bayesian Gaussian process, following what has been presented in Chapters 3 and 4. The use of a full Bayesian Gaussian process provides two advantages in history matching applications. Firstly, uncertainty in the surrogate itself is mitigated due to the marginalisation of the Gaussian process' hyperparameters. Secondly, as a by-product of Gaussian process emulators, code uncertainty can be directly estimated due to the analytical expression for the output variability, as mentioned in Chapter 3.

Using a Gaussian process emulator and taking into account both the model discrepancy and measurement error variances, the Bayesian history matching procedure discards regions of the input space in an iterative scheme where each iteration is referred to as a *wave* [216]. Let  $\bar{I}(\mathbf{x})$  denote the implausibility measure of the input configuration  $\mathbf{x}$  given the observed datum  $z$ . The implausibility is defined as

$$\bar{I}(\mathbf{x}) = \frac{|z - \mathbb{E}[f(\mathbf{x})]|}{\sqrt{\sigma^2(\mathbf{x}) + \sigma_{\text{md}}^2 + \sigma_{\text{me}}^2}}, \quad (5.1)$$

where  $\mathbb{E}[f(\mathbf{x})]$  is the mean response of the emulator and  $\sigma^2(\mathbf{x})$  the emulator's predicted error estimate. It is important to note that in certain applications the simulator output is known to be stochastic and an additional term is added in the denominator to account for *ensemble variability* [3]. In this work such variability is not needed. Furthermore, note that the distance between the data threshold  $z$  and the surrogate output is standardised by the sum of the modelled uncertainties. The implausibility function allows one to identify which configuration points are

far from the target by measuring the number of standard deviations. In the literature, Pukelsheim's three sigma rule [182] is a common choice to characterise the number of standard deviations. The rule is stated as follows

**Theorem 5** (Pukelsheim's three sigma rule). *Let  $X$  be a real random variable with mean  $\mu$  and variance  $\sigma^2$ . Let us assume that  $X$  follows a uni-modal distribution. The three sigma rule states that the probability for  $X$  falling away from its mean by more than 3 standard deviations is at most 5%. That is,*

$$P\{|X - \mu| \geq 3\sigma\} < 0.05 \quad (5.2)$$

For this end, the regions of input configuration space where the surrogate should refocus is then defined as

$$\mathcal{X}_{\text{NROY}} = \{\mathbf{x} \in \mathcal{X} : \bar{I}(\mathbf{x}) < a\}. \quad (5.3)$$

The subscript stands for the *Not-ruled-out-yet* (NROY) [217]. The threshold level  $a$  is set to 3, as it includes 95% of the probability mass for a unimodal univariate distribution.

### 5.1.1 Probabilistic History Matching

The previous description of history matching stems from the construction of probability by using mathematical expectation as a primitive. This is known as Bayes linear statistics, since linearity of expectations is a key aspect in the development of tools under the theory. For a more thorough discussion on Bayesian linear methods refer to [100].

The construction of the implausibility measure under Bayes linear can be seen as a function that uses a pointwise estimate from the surrogate. Nonetheless, the Gaussian process provides an emulator, a probabilistic generator of surrogate models, which can be exploited in a full probabilistic formulation. As discussed in previous chapters, the Gaussian process itself depends on its own set of hyperparameters. Thus, a full probabilistic approach is proposed in this chapter. Both the Gaussian process probabilistic output is considered for an emulator, and the posterior distribution of the hyperparameters is also taken into account.

In the proposed characterisation, a full probabilistic treatment of the implausibility function is used. This is achieved by incorporating the probabilistic distribution

of the Gaussian process emulator's output. Let the implausibility random variable be

$$I(\mathbf{x}) = \frac{|z - f(\mathbf{x})|}{\sqrt{\sigma_{\text{md}}^2 + \sigma_{\text{me}}^2}}, \quad (5.4)$$

where  $f(\mathbf{x}) \sim \mathcal{N}(m(\mathbf{x}), \sigma^2(\mathbf{x}))$  is the Gaussian process emulator for the simulator output. The NROY space is thus characterised by probabilistic statements of the form

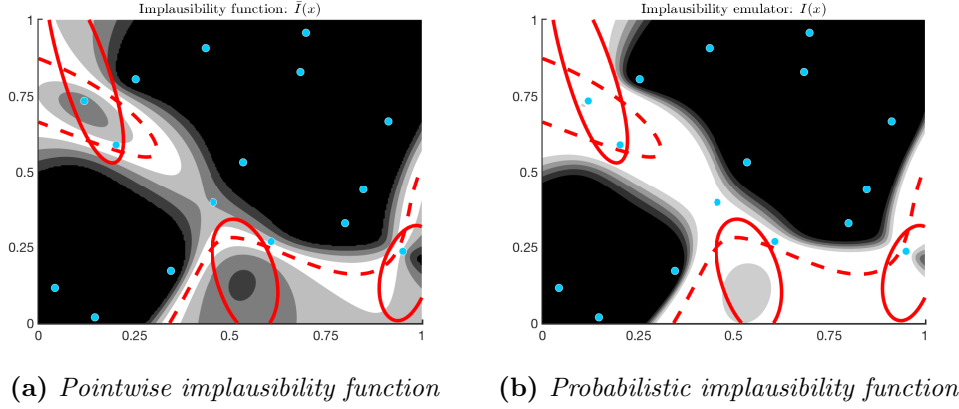
$$P\{I(\mathbf{x}) \leq 3\}, \quad (5.5)$$

which is analogous to the probabilistic statements of Holden et al. [117], and Williamson et al. [224].

The difference between each implausibility function, the pointwise estimate and the probabilistic characterisation is shown in Figure 5.1. In both, dark shaded regions correspond to high values of implausibility, whereas light shaded regions account for low values. Pitch-black regions in 5.1a indicate values of the implausibility function of 3 or more, whereas in 5.1b it indicates a probability of implausibility less than that of machine precision ( $\approx 10^{-12}$ ). This allows one to be confident that the desired output level won't be achieved in such dark regions. It can be noted how the pointwise implausibility (5.1) tends to follow the surrogate output while the probabilistic implausibility (5.4) covers a wider region. This is typically known as the exploration-exploitation trade-off in computer experimental design [80]. Exploration is desirable as the use of a surrogate model might induce some bias if followed too blindly in the first steps of the procedure.

## 5.2 NROY space identification

History matching relies on the correct identification of the region of input space  $\mathcal{X}$  where the simulator is likely to replicate the measured data. At every iteration, the NROY space becomes orders of magnitude smaller than the original space, and can exhibit a complex or disconnected topology. As a result, naive rejection-based sampling can quickly become highly expensive. To address this deficiency, different alternatives have been devised in the literature. Williamson and Vernon [223] proposed an implausibility driven sampling scheme which needs to define an



**Figure 5.1:** Illustration of the contour levels for the implausibility function around a target level (solid red line). The Gaussian Process emulator provides the dashed lines as its predictions. The left panel shows the implausibility measure using a pointwise estimate. The right panel shows the mean implausibility function derived from stochastic surrogate. It can be seen that the stochastic version achieves better coverage than the pointwise alternative.

appropriate threshold ladder. Yeh et al. [225] use clustering to identify possibly disconnected regions. Andrianakis et al. [3] proposed to use Gaussian random variables centred at the mean from the NROY points of wave  $t$  to generate points for wave  $t + 1$ . For this, the covariance matrix is chosen so that much of the input space can be covered, ideally accepting 20% of the proposed samples.

At present, the correct identification of the NROY space in a full probabilistic setting has not been fully explored. To address this limitation, this chapter proposes the use of a sampling scheme that is both able to overcome complex topologies and able to generate uniform samples from the target region. Inspired by sequential Monte Carlo, simulated annealing and subset stochastic optimisation, Beck and Zuev [14] developed an algorithm (AIMS) that samples uniformly on a set of interest. In [226], the algorithm (AIMS-OPT) was shown to achieve excellent results in complicated optimisation settings, *e.g.* when the maximum can be achieved in a ridge on input space. This motivated the development of the TA<sup>2</sup>S<sup>2</sup> algorithm in Chapter 4, where a modification was proposed to improve the efficiency by means of a slice sampling component, as well as exploiting its parallelising capabilities.

In this work, the focus is on regions in which the probability of non-implausibility



in input space  $\mathcal{X}$  is maximal. That is, we aim for the maximisers of  $P\{I(\mathbf{x}) \leq 3\}$ , which are hopefully close to the upper bound of 1. This objective corresponds to the three sigma rule, which states that 95% of probability is achieved within three standard deviations from the mean. By means of the TA<sup>2</sup>S<sup>2</sup> algorithm in [84] a nested sequence of sample sets  $U_m \subset \dots \subset U_0$  is obtained such that

$$U_j = \left\{ \mathbf{x}_i^{(j)} : \mathbf{x}_i^{(j)} \sim p_\tau(\mathbf{x}), i = 1, \dots, N \right\}, \quad (5.6)$$

where  $p_\tau$  denotes an intermediate density that converges to a uniform density in the set of optimisers, and  $N$  is the number of samples extracted at every level of the annealing schedule [see 84, for more details].

One of the advantages of using TA<sup>2</sup>S<sup>2</sup> in history matching is that if the resulting set  $U_m$  is highly concentrated at one probability level the previous level of samples can be used instead for exploration. This would be appropriate, if more samples from lower probability responses are needed. In Figure 5.2 an application of the TA<sup>2</sup>S<sup>2</sup> algorithm is shown to adaptively identify the NROY space for a torus example presented in [223]. The function is defined over the 3-dimensional cube  $[-20, 40]^3$  as its expression is as follows. Let  $\mathbf{x} = (x_1, x_2, x_3)^\top$ , and define the 2-dimensional projection as

$$\mathbf{u} = \begin{bmatrix} (x_1 - 2)^2 - 3 \\ (x_2 - 2)^2 - 3 \end{bmatrix}, \quad \Sigma = \frac{1}{2^{12}} \begin{pmatrix} 1 & -0.97 \\ -0.97 & 1 \end{pmatrix}. \quad (5.7)$$

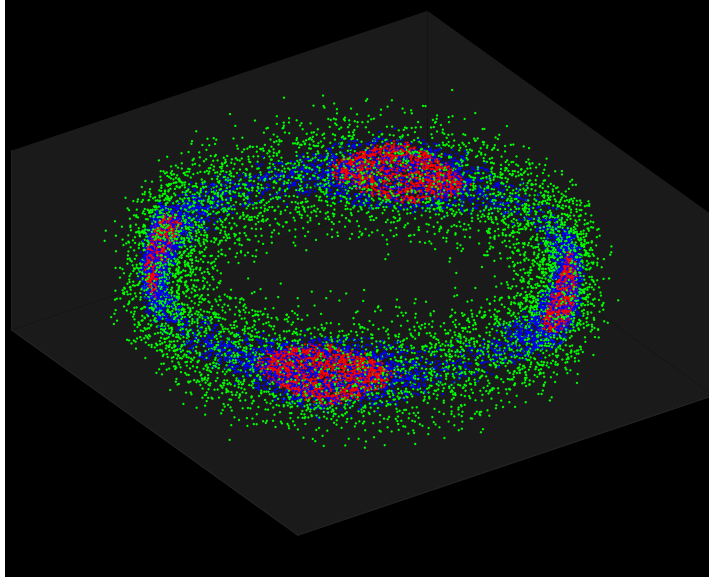
The implausibility function for this numerical exercise is defined as

$$I(\mathbf{x}) = \frac{1}{10} \left( \sqrt{\mathbf{u}^\top \Sigma^{-1} \mathbf{u}} + \frac{x_3^2}{0.04^2} \right). \quad (5.8)$$

Therefore, the implausibility function induces the 4 the modes in a torus as shown in Figure 5.2. For this numerical exercise, the only assumed source of uncertainty is the measurement error as the exact model is being used in the search, *i.e.* no emulator was used as a surrogate.

## 5.3 Sequential non-implausible design

Having successfully identified the NROY space, the question of how to query from such region to refocus the emulator remains a challenging problem. The



**Figure 5.2:** *Samples being generated at each wave for the torus implausibility function of Williamson and Vernon [223]. At the final level, the adaptive sampler correctly identifies the four regions of the zeroth contour level (in red). It should be noted that exploration of the space could be done with samples from previous levels if required.*

seminal papers [48] and [49] performed a sequential design for this purpose. The common choice in the literature is to select points greedily from the NROY space to build a new emulator completely focused on that region [191]. This implies that a new emulator is built based on the identified region at every wave. In this dissertation, a different (albeit conservative approach) is followed. Since the simulator is assumed to be computationally very expensive, it might seem unrealistic to expect that the computational budget is kept the same at every *wave*. Moreover, discarding points might represent a waste of resources and information. In turn, the points must be chosen carefully at each wave to later add them to the set of training runs and build a new Bayesian emulator.

The idea is that the most general information would likely be extracted at the very first iterations while greater accuracy will be pursued at later stages of the history matching procedure. For example, at the first waves a good characterisation of the global trend can be identified. It is therefore appropriate to guide the choice of training points by following a learning criteria. This is done in Bayesian optimisation or in reliability analysis problems [218, 19, 139]. For comparison, three active learning criteria are presented. The first criterion is the expected

contour improvement by Ranjan et al. [184]. The improvement function is defined as

$$I(\mathbf{x}) = \epsilon^2(\mathbf{x}) - \min \{ (f(\mathbf{x}) - z)^2, \epsilon^2(\mathbf{x}) \}, \quad (5.9)$$

where  $f(\mathbf{x})$  is the Gaussian process emulator at configuration  $\mathbf{x}$ ,  $z$  the targeted contour level, and  $\epsilon(\mathbf{x}) = k v(\mathbf{x})$  the number of predicted standard deviations derived from the uncertainty model,  $v(\mathbf{x}) = \sqrt{\sigma^2(\mathbf{x}) + \sigma_{\text{md}}^2 + \sigma_{\text{me}}^2}$ . In history matching  $k$  is typically set to 3, following the three sigma rule. As the emulator is random in nature, the expected value of the contour improvement is used instead of a pointwise estimate. The expected contour improvement (ECI) can be computed as

$$\begin{aligned} \mathbb{E}[I(\mathbf{x})] &= [\epsilon^2(\mathbf{x}) - (\mu(\mathbf{x}) - z)^2 - \sigma^2(\mathbf{x})] [\Phi(u_2) - \Phi(u_1)] \\ &\quad + \sigma^2(\mathbf{x}) [u_2 \phi(u_2) - u_1 \phi(u_1)] \\ &\quad + 2[\mu(\mathbf{x}) - z] \sigma(\mathbf{x}) [\phi(u_2) - \phi(u_1)] \end{aligned} \quad (5.10)$$

where  $\epsilon(\mathbf{x})$  is the  $k$ -th multiple of standard deviations,  $u_1 = (z - \mu(\mathbf{x}) - \epsilon(\mathbf{x}))/\sigma(\mathbf{x})$ ,  $u_2 = (z - \mu(\mathbf{x}) + \epsilon(\mathbf{x}))/\sigma(\mathbf{x})$ , and  $\Phi(\cdot)$  and  $\phi(\cdot)$  are the standard Gaussian cumulative and density functions respectively.

The second criteria to be tested is the expected risk by Echard et al. [73], which was originally was designed for reliability analysis. This implies learning the set  $\{\mathbf{x} : g(\mathbf{x}) > 0\}$ , with  $g(\mathbf{x})$  the performance or limit-state function of a configuration  $x$ . The critical level  $g(\mathbf{x}) = 0$  is referred to as a *transition* level, as its correct emulation will allow one to effectively classify a given configuration in the simulation in terms of the system's performance. In this chapter, the problem is explicitly stated in terms of the contour level  $z$  which corresponds to the observed data in the experimental setting. This means that the risk function is defined as

$$R_z(\mathbf{x}) = \begin{cases} (f(\mathbf{x}) - z)_+ & \text{if } \mu(\mathbf{x}) \leq z \\ (z - f(\mathbf{x}))_+ & \text{if } \mu(\mathbf{x}) > z \end{cases}, \quad (5.11)$$

where  $(\cdot)_+$  denotes the non-negative part of the argument, and  $\mu(\mathbf{x})$  denotes the expected value of the Gaussian process emulator at configuration  $\mathbf{x}$ . The expected risk is used as a learning criteria due to the random nature of the output of the emulator. The derivation is a straightforward solution of one dimensional

Gaussian integration, which for completeness is included in 5.A. The analytical expression can be written in compact form as

$$\mathbb{E}[R_z(a)] = \sigma(\mathbf{x}) [-\text{sign}(\bar{z}) \bar{z} \Phi(-\text{sign}(\bar{z}) \bar{z}) + \phi(\bar{z})], \quad (5.12)$$

where  $\bar{z} = (z - \mu(\mathbf{x}))/\sigma(\mathbf{x})$  denotes the standardised contour level,  $\text{sign}(\cdot)$  the sign function, and the pair  $\Phi(\cdot)$  and  $\phi(\cdot)$  are the cumulative and density functions of a one-dimensional Gaussian random variable respectively.

Lastly, the third learning criteria to be compared, is a variation of the entropic profile presented by Lv et al. [145]. Originally formulated in the reliability analysis literature, it was designed to measure the entropy of a random variable in a neighbourhood of two standard deviations from the origin. In this chapter the concept has been extended. Again, an explicit solution is presented for a contour level  $z$  observed in the experimental data. The entropic profile is defined as

$$H_z(f(\mathbf{x})) = \left| \int_{z-k\sigma(\mathbf{x})}^{z+k\sigma(\mathbf{x})} -\ln \pi(y) \pi(y) dy \right|, \quad (5.13)$$

where  $\sigma^2(\mathbf{x})$  denotes the predicted variance from the emulator and  $\pi(y)$  the non-standard Gaussian distribution from the emulator response  $y = f(\mathbf{x})$  is distributed as a  $\mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x}))$ . As shown in 5.B, the entropic profile can be written compactly as

$$H_z(\eta(\mathbf{x})) = \left| \left[ \ln \left( \sqrt{2\pi} \sigma(\mathbf{x}) \right) + 0.5 \right] [\Phi(\bar{z}_2) - \Phi(\bar{z}_1)] - 0.5 [\bar{z}_2 \phi(\bar{z}_2) - \bar{z}_1 \phi(\bar{z}_1)] \right|, \quad (5.14)$$

where  $\bar{z}_1$  and  $\bar{z}_2$  denote the standardised contour levels, that is,  $\bar{z}_1 = (z - \mu(\mathbf{x}) - k\sigma(\mathbf{x}))/\sigma(\mathbf{x})$  and  $\bar{z}_2 = (z - \mu(\mathbf{x}) + k\sigma(\mathbf{x}))/\sigma(\mathbf{x})$ .

All three of the previously discussed criteria rank the samples from the identified NROY space. It is important to note that these type of learning criterion take into account a one-step-look-ahead pointwise strategy. Other options include A-optimal designs, which incorporate area impacts to the improvement of the emulator's response surface. In particular, following dynamic programming strategies, one can define a learning criteria with a known number of sequential decisions. As a consequence, this selection of points chooses a batch of candidate runs. This is known as finite-horizon dynamic programming [28].

Given the proposed learning criteria, a natural question is how to choose the points given such ranking. As nearby sample points are likely to be similarly ranked, it would be naive and a waste of computational resources to query the simulator on just the top-ranked NROY samples. Alternatively, to choose uniformly from the sample points would ignore any ranking at all. An adequate leverage between these extremes is achieved by the following procedure.

Firstly, a learning criterion is chosen to rank the samples, as well as a cut off point. That is, a threshold to guarantee that some percentage of the maximum attainable gain can be held. In this work, this percentage is assumed to be 50%. The current setting selects the best first point following an active learning criteria. Although, the rest of chosen points will have a diluted effect from the learning criteria. As commented above, choosing a high percentage would likely concentrate the candidates in narrower neighbourhoods around the best sample. In contrast, a low percentage would not acknowledge the ranking. Secondly, a maximin design is proposed to choose the next batch of training points as follows. The seed of this design is selected to be the top ranking point from the NROY samples, *i.e.* the point that has the highest expected learning criteria. It is important to note that this type of sampling usually starts with the mean of a cloud of points [see 139, for more details]. Since the NROY space has potentially a complicated topology, the average point might lie outside the region of interest and choosing a point from outside the NROY space would be a poor selection.

After choosing the starting point  $\mathbf{x}_0$ , the maximin design proposes the furthest sample available, so  $\mathbf{x}_1 = \arg \max_{\mathcal{X}_{\text{NROY}}} \|\mathbf{x} - \mathbf{x}_0\|$  is chosen as a successor. This selection takes both points out from the bag of samples and initialise the set of new sample points containing both,  $\mathcal{X}^* = \{\mathbf{x}_0, \mathbf{x}_1\}$ . The training points currently in use for the emulator are also included in  $\mathcal{X}^*$ , that is  $\mathcal{X}^* = \mathcal{X}^* \cup \mathcal{D}$ . This prevents redundancy in the selection of new points. The next step computes the distances to both  $\mathbf{x}_0$  and  $\mathbf{x}_1$  for the remaining sample points in the sampled NROY collection. After this, each sample point in the NROY set has two associated distances to  $\mathcal{X}^*$ . The next step involves choosing the minimum for each NROY sample. Thus, the distance of each point to the set  $\mathcal{X}^*$  is computed. Following this,  $\mathbf{x}_3$  is assigned as the furthest point observed and is taken out of the bag of samples and incorporated to the collection of new training runs. That is, the update  $\mathcal{X}^* = \mathcal{X}^* \cup \{\mathbf{x}_3\}$  is performed. The selection of new points follows this procedure iteratively, computing the distances to the set  $\mathcal{X}^*$  and choosing  $\mathbf{x}_k$

as the farthest point. Performing the previous steps allows us to obtain  $N$  new samples to train a refocused emulator. As mentioned above, the first point will follow the specific learning criteria chosen. The rest of selected points will followed a space-filling design with a diluted effect from the criteria in use.

In summary, the use of the maximin selection allows us to (i) retain the best possible point; (ii) collect samples from a space-filling design in NROY space; and (iii) restricts the choice of new points following the active learning criteria.

## 5.4 Numerical experiments

In this section a comparison among the active learning criteria is presented. The performance of the history matching procedure is illustrated in a 2D example, a 3D case study of a fault model, and on battery of tests in multiple dimensional settings.

In history matching applications, a common stopping rule to terminate the procedure is to compare the maximum predicted error of the surrogate model in the NROY samples to the estimated variance attributed to measurement and model discrepancy ( $\sigma_{\text{me}}^2$  and  $\sigma_{\text{md}}^2$ ). The motivation is that further improvement of the surrogate would not be able to reduce the elicited deviations from the simulator. The Gaussian process, in this setting, is ideal since it provides an estimation of predicted error as by product in its construction.

As an alternative to the maximum criteria discussed above, the use of a scoring rule is explored in this work. For these purposes the Continuously Ranked Probability score (CRPS) presented in Chapter 4 is used. It possesses the properties of being a proper scoring rule to report probabilistic inferences. As stated before, the Gaussian process emulator is able to provide full probabilistic statements like predicted values and dispersion estimates around such predictions.

### 5.4.1 Franke's function

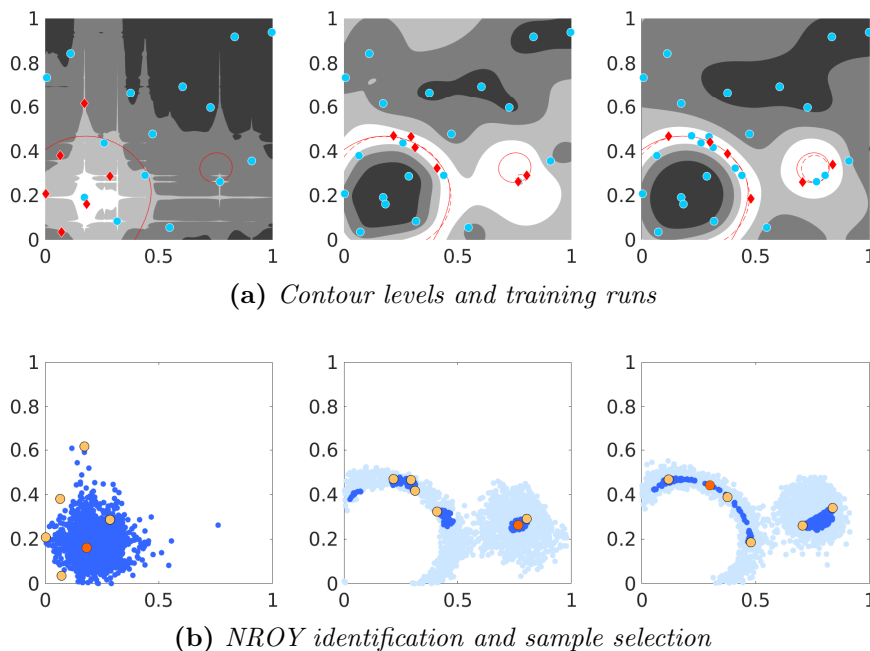
In this experiment, Franke's function is used as a simulator [82]. As mentioned in Chapter 4, it is a well known function in the surrogate modelling literature. It is defined in the two dimensional unit cube, and consists of a sum of three Gaussian peaks and one smaller dip. Franke's function is defined as

$$f(\mathbf{x}) = 0.75 \exp\left(-\frac{(9x_1 - 2)^2}{4} - \frac{(9x_2 - 2)^2}{4}\right) + 0.75 \exp\left(-\frac{(9x_1 + 1)^2}{49} - \frac{9x_2 + 1}{10}\right) \\ + 0.5 \exp\left(-\frac{(9x_1 - 7)^2}{4} - \frac{(9x_2 - 3)^2}{4}\right) - 0.2 \exp(-(9x_1 - 4)^2 - (9x_2 - 7)^2). \quad (5.15)$$

As mentioned before, Figure 4.1 depicts the contour levels of Franke’s function. For the purposes of history matching, the target contour level has been defined as  $z = 0.6$ , which results in a target contour level with two disconnected disks, as shown as a solid red line in Figure 5.3a. The dashed red lines correspond to the emulator’s predicted contour level of interest. The shaded regions represent the probabilistic implausibility contour levels, lighter colours denote a higher probability. The apparent discontinuities of the shaded regions are due to both the low number of training runs and the emulator hyperparameters being sampled from the posterior distribution. As previously discussed, when the available data to train an emulator is small, multimodal samplers are able to represent code uncertainty more robustly, [see 84, for more details on multimodal posterior samplers for Gaussian processes].

Panels in 5.3a show the sequences of waves of the history matching procedure. The scatter points in blue represent training points at that wave used to fit the emulator, whereas red dots depict the chosen points in the identification by active learning. For the purposes of illustration, the entropic profile discussed in Section 5.3 was chosen. The samples generated from the NROY space at each wave are shown in Figure 5.3b. In this case, the use of sampling algorithms based on annealed distributions is justified by the complex geometry of the target region [226, 84]. In particular, the first panel shows that all NROY samples satisfy the property of being good candidates to improve the emulator. In the same panel, orange dots denote the chosen points after selecting the top ranking sample. For the remaining subpanels, light blue dots illustrate sample points from the NROY space which are not suitable to improve the emulator. The best candidates are depicted with dark shaded blue following the ranking of the active learning criteria and a selection of a threshold. Also, the space filling interpretation of the maximin strategy is demonstrated empirically in the first panel (the first wave of history matching). As noted before, good coverage can be seen in all panels by the maximin space filling criteria.

The procedure was replicated by 50 independent runs of the procedure for



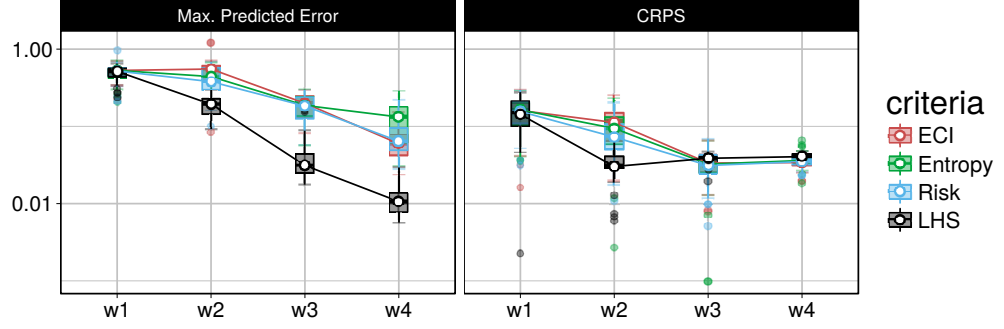
**Figure 5.3:** Results for Franke's function in the history matching setting. In 5.3a, contour levels of the probabilistic implausibility measure are shown with lighter shades representing higher probabilities. Blue dots represent training runs used for the simulator at wave  $t$ , and red dots new points identified in NROY space with good predicted improvement performance. Each subpanel in 5.3b shows the samples in NROY space, with those satisfying a good predicted improvement in darker colours. In red, points selected to try the simulator to train the Gaussian process emulator are shown.

each learning criteria. This is depicted in Figure 5.4, where results are shown for both the maximum predicted error and the CRPS as iterations advance. The predicted errors steadily decrease for each learning criteria. However, it can be seen that the decrements in CRPS are not constant. The LHS criterion chooses among the NROY samples by the maximin design without any ranking or prescribed threshold. It is important to note that the LHS sampling scheme seem to decrease the predicted error too hastily. However, it becomes trapped in its own overconfidence as seen by a slight increase in the CRPS.

### 5.4.2 The IC fault model

The following experiment tests the emulation-based history matching framework in a physical model. The IC fault model is a cross-sectional simulator of a reservoir.





**Figure 5.4:** *History matching for Franke's function. The procedure was performed independently 50 times for each active learning criteria. The results are summarised in boxplots at each wave. The Maximum predicted error was calculated from the NROY samples at each wave. Analogously, the reported CRPS was computed as the median from the NROY samples. Although a space-filling criteria reduces the predicted error further than the other candidates, the probability statements seem to deteriorate when compared to the other learning criteria.*

Each run is determined by three unknown input parameters, namely,  $h$  (the fault throw),  $k_g$  (the good-quality sand permeability) and  $k_h$  (the poor-quality sand permeability) [212, 191]. It has become a benchmark to test history matching procedures as it is a difficult computer model to calibrate.

The outputs of this simulator are 36-month time series corresponding to three different properties such as the oil production rate, the water injection rate and the water production rate. The information of this model is stored as a collection of 159,661 code runs selected uniformly at random in the 3-dimensional cube. Instead of matching the full time series only three statistics are chosen as in [191]. Those outputs are  $o_{24}$  the oil production rate at month 24;  $o_{36}$  the oil production rate at month 36; and  $w_{36}$ , the water injection rate at month 36.

For experimentation purposes it is assumed that there is no access to such a rich dataset. In turn, a handful of 60 points are chosen at random from a Latin hypercube sampling scheme to initialise the procedure. At each wave an additional 30 points are selected as discussed in Section 5.3 to improve the emulator at the pre-specified target level. Each output is emulated independently by a Gaussian process. In this case, the implausibility function is a probabilistic version of the Second Maximum Implausibility Measure Vernon et al. [216], computed from the GP posterior distribution using a Monte Carlo estimate. This implausibility is

used in order to guard against the possibility that one of the emulators is not performing accurately. The Second Implausibility Measure is defined as

$$I^{(2)}(\mathbf{x}) = \max_i (\{I_{(i)}(\mathbf{x})\} \setminus I^{(1)}(\mathbf{x})), \quad (5.16)$$

where  $I_{(i)}$  denotes the implausibility for the  $i$ -th output, and  $I^{(1)}$  denotes the largest Implausibility among all outputs. The target level to be matched is defined as

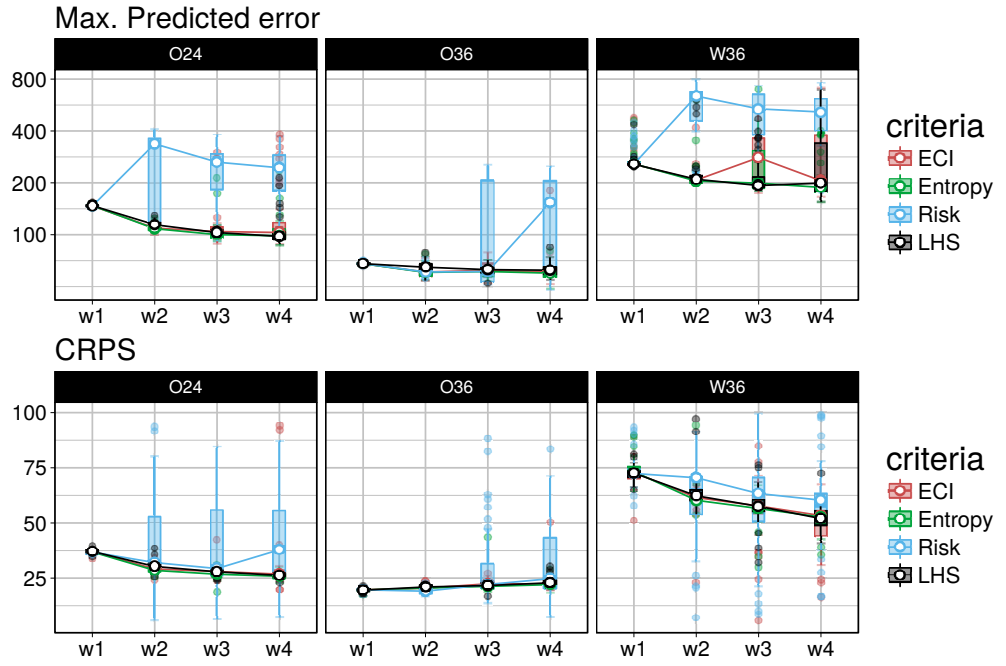
$$\mathbf{z} = (563.6, 387.5, 917.2)^\top. \quad (5.17)$$

The history matching procedure was replicated 50 times, each starting with a different LHS design, and both performance measures for each wave were recorded. Results are summarised as boxplots in Figure 5.5. It is clear that the expected risk learning criteria is both slower and leads to noisier results for this simulator. In all cases, the oil production rate at month 36, *i.e.*  $o_{36}$ , proves too difficult to emulate as seen from the boxplots in Figure 5.5. Nonetheless, history matching overcomes this limitation and manages to decrease both the expected predicted error and probabilistic predictions in the target contour level defined (5.17). It is important to note that better results can be achieved if a different GP emulator is trained at every contour level as in the spirit of [191]. This work focuses on the properties of using both the complete probabilistic statements from the Bayesian posterior of the computer code and the incorporation of active learning criteria that uses this characterization.

### 5.4.3 Random functions experimental setting

In order to assess the impact of each learning criteria within the full Bayesian history matching procedure, the following experimental set-up is proposed. It is inspired by [115], as it was used to measure the performance of different acquisition functions used for Bayesian optimisation. The reason for following this direction is that there is no generally-agreed set of test functions for high dimensions.

Different dimensional settings are chosen in order to understand both limitations and strengths of the three active learning criteria for one dimensional output codes. The LHS discussed in the previous experiments is included in the comparison. Since there is no current battery of test functions available, it is proposed that the emulation-based history matching is applied to random functions generated



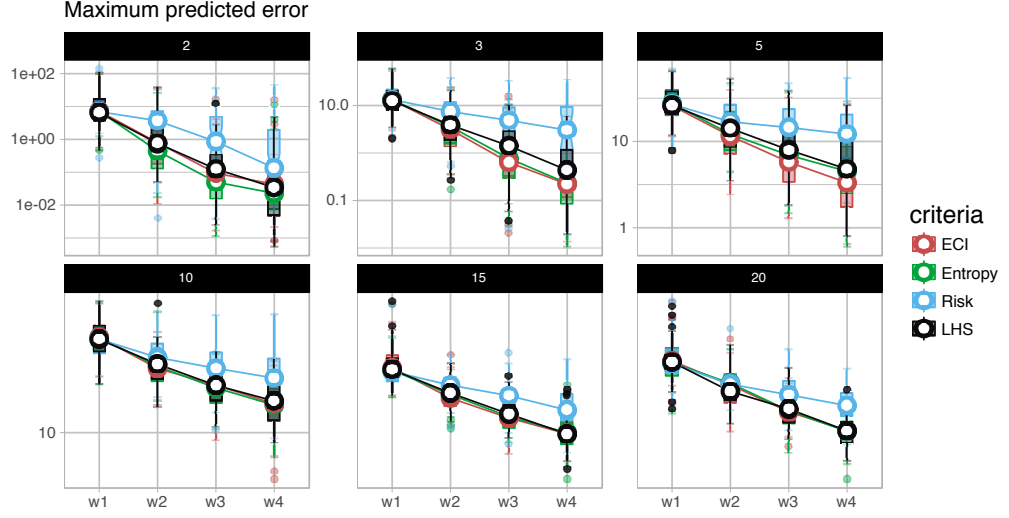
**Figure 5.5:** Results for the IC fault model in both performance measurements. Each box corresponds to each output from left to right. By using the probabilistic Second Maximum Implausibility measure it can be noted that there is no bias towards an inaccurate emulator. There is evidence that the oil production rate at month 36,  $o_{36}$ , proves difficult to fit with the chosen GP assumptions. Nonetheless, the history matching procedure is reducing both the uncertainty and the prediction error for the target contour level.

from a Gaussian process prior, as shown in 5.C. Having different random seeds, it is possible to replicate the same function for each active learning criteria in each dimensional setting, thus preserving each set of the functions to be compared. In total, 50 random functions were simulated in each dimensional setting. For each, the contour level corresponding to the 95% percentile on the prior seeds as explained in 5.C is chosen as the target for the History matching procedure.

The different dimensional settings include low dimensional spaces (2D and 3D), medium-sized dimensional spaces (5D and 10D) and large dimensional spaces (15D and 20D). In the case of large dimensional settings, dimensionality reduction techniques can be applied such as active variable selection [217] or Partial Least Squares [35]. It is widely known that the use of Gaussian process as surrogate models tend to lose predictive accuracy and robustness with increasing dimensionality. This happens because of the Kernel being used for the correlation structure, as typical choices rely on some form of Euclidean distance. Thus, the dimensional settings chosen reflect the feature spaces where the GP emulator will be able to generalize well.

In the experiments, the history matching is not terminated, but the predicted error is tracked along the iterations of the procedure. Figure 5.6 depicts the maximum predicted error is reported at every wave, for each random function. The results are grouped in boxplots to show the overall dispersion at each wave. The medians are connected with lines between waves for every learning criteria for visualisation purposes. The space filling baseline results are shown in black. Overall, the history matching procedure is successful in reducing the maximum predicted error. It is important to note that in any dimensional setting the Risk criterion in (5.12) shows less improvement as the waves advances. This is a consequence of the Risk criterion being more susceptible to local exploration than the other candidates. In low-dimensional settings the Entropic profile and the ECI show a slight advantage over the space filling design. This is a consequence of a better leverage between the exploitation and exploration trade-off. In high-dimensional settings, both the Entropic profile and the ECI show comparable performance to that of the space filling design. This is evidence that both criteria are being too general in their rankings and little exploitation of the surrogate is being used.

The reduction of the CRPS by the emulation-based history matching is shown in Figure 5.7, again by the trend in the lines that connect the waves of the procedure.

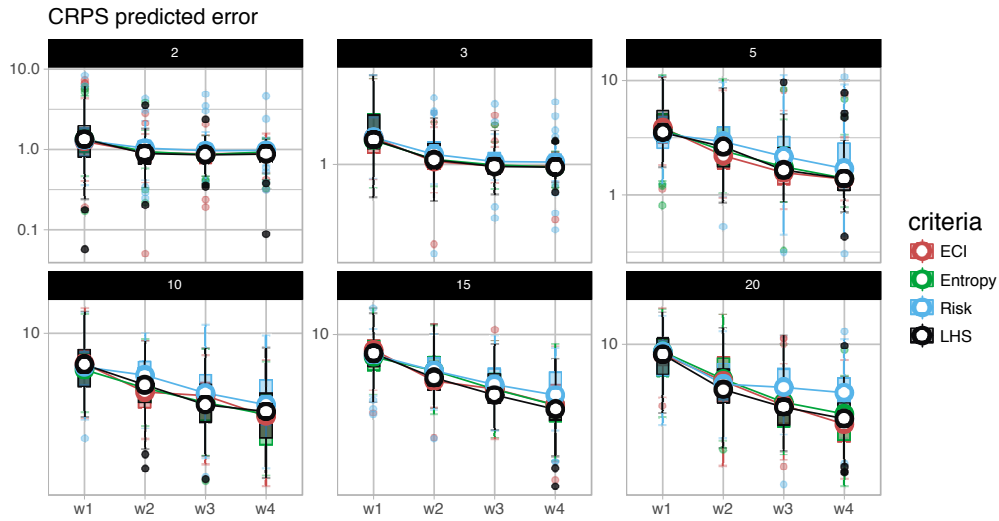


**Figure 5.6:** Results for the maximum predicted error in NROY samples. Each subpanel corresponds to different dimensional settings indicated in the black headers. The boxplots are generated by extracting the statistic from each of the replications of the experiment at each wave. Connecting lines are shown to better appreciate the downward trend as the iterations succeed. In all dimensional settings the risk learning criteria is confirmed to be slowest as in Figure 5.5.

As before, the overall performance is as desired, resulting on decreasing values of the score. The use of the Risk learning criterion seems to be hindered again by its lack of willingness to explore the NROY space as the dimension of the problem increases. As before, the entropic profile and the ECI show comparable results to a space filling design.

The results show that the Risk learning criterion (5.12) is prone to get being trapped in local regions around the best choice. In contrast, for low and medium dimensional settings, the Entropic profile and the ECI show a better performance in lowering the maximum predicted error (the variance estimated by the emulator).

For higher dimensional settings, there is no apparent gain in using any of the criteria discussed above. The use of a space-filling design seems like a safe choice. However, it should be noted that the learning criteria do achieve a lower predicted error in low dimensional settings. This should be taken as an indication that something can be done to enhance the performance on higher dimensional problems. Recall that the selection of the samples to refine the emulator is done in two stages. Firstly, the NROY space is identified by an annealed uniform sampling



**Figure 5.7:** Results for the CRPS predicted error in NROY samples. Each subpanel corresponds to different dimensional settings indicated in the black headers. The boxplots are generated by extracting the statistic from each of the replications of the experiment at each wave. In this case the median of the CRPS is extracted from the NROY samples. Connecting lines are shown to better appreciate the downward trend as the iterations succeed. It should be noted that very low dimensional setting the prediction on the target level do not improves substantially. However, for larger dimensional spaces it continues to improve.

scheme. Secondly, the samples are ranked accordingly (choosing a learning criteria) and those that are not able to produce at least a 50% improvement than the best in the batch are discarded. Following this, among the samples retained, a minimax selection procedure is done, starting by the top sample. From the results previously exposed, it seems that setting this 50% target level seems too permissive and that most of the samples are retained in the procedure. In the end, the minimax selection and a space-filling choice become equivalent. This is another embodiment of the curse of dimensionality, in which higher dimensionality requires larger the sampling designs for the emulator. An alternative, which is subject of current research, is to choose a batch of good candidates from the learning criteria as it is done in the Bayesian optimisation setting with the multi-point expected improvement from Chevalier and Ginsbourger [43].

## 5.5 Discussion

In this chapter, it has been proposed to completely acknowledge the information of the Gaussian process surrogate in history matching applications. This leads to the use of a full probabilistic version of the implausibility function. The exploitation of this measure is done by sampling with an annealing schedule as in sequential subset optimisation. This allows one to generate uniform samples from the regions defined by a high probability of being non-implausible. In these regions, the simulator is likely to replicate the observed limited data. There ability to sample from complicated geometries and disconnected regions is achieved by using this form of annealed sampling. The sampling methods have been recently proposed in the Bayesian inference framework but are flexible enough to accommodate to the history matching setting [84]. Additionally, the use of active learning criteria to improve the surrogate was also presented. The experimental results show evidence of better performance when using the expected contour improvement or the proposed entropic profile, a version adapted to history matching. This contrasts with random generation of samples by some type of adapted proposals or rejection-based methods. A testbed of random functions was presented to test the effectiveness of this framework, since there is no agreed collection of history matching test functions. It is important to note that the learning criteria used in this work can be classified as myopic. This means that the learning functions only take into account the information available at the current iteration. The learning criteria can potentially decrease the number of samples to be considered and achieve comparable results to that of using the whole set of points, as in LHS. Note that theoretical results in [208] show that the GP emulator converges to the true simulator with rates depending on the coverage of the training point design. The experimental results here suggest that certain alternatives can achieve similar consistency. Also, the results shown for the IC-Fault model enhances the need to study further multi-output history matching application. A direction of current research is the use of more general learning criteria, or acquisition functions, that mimic batch optimisation. In the following section an industrial application of the emulation-based history matching is discussed.

## 5.6 Industrial application

This problem was an uncertainty challenge posed by Airbus at the *Uncertainty Quantification and Management* (UQ&M) study group hosted by the Institute for Risk and Uncertainty at the University of Liverpool on July 2016. The objective of this exercise was to narrow the set of feasible aircraft configurations that matched a specific competitive advantage in a *realistic but not real* environment in a working session of three days. In particular, the objective was to satisfy certain threshold targets in flyover and sideline noise, Nitrogen Oxide emission and fuel consumption while satisfying certain tolerance levels. Problems of this type are known in the literature as Robust-based Design Optimisation (RBDO) [65, 38].

The model consisted of a 24-hour operation aircraft, and was treated as a black-box function implemented in the AirCADia framework [106] due to a disclosure agreement. This model is a multi-dimensional simulator in both input and output spaces. The input parameters are summarised in Table 5.1 and represent design variables, and are referred to as input variables in the context of simulator code analysis. The output variables represent figures of merit for the aircraft to satisfy and are summarised in Table 5.2. The RBDO formulation of the problem offers an analogue to the history matching setting. The latter is used to discard regions of input parameter space in order to satisfy target values of the merit functions given certain variability levels. These variability thresholds are treated as observational noise in the process, allowing the correspondence between calibration under the history matching formulation and that of RBDO to be completed.

Notation	Range	Description
WA	[1300, 1400]	wing area
WAR	[9, 11]	wing aspect ratio
ST	[26000, 32000]	sea level static thrust
FPR	[1.5, 1.8]	fan pressure ratio
OPR	[30, 40]	overall pressure ratio
B	[6, 8]	bypass ratio

**Table 5.1:** *Input variables for the Airbus application.*

The problem solved during the study group is described as follows. The target design is for that of an aircraft that satisfies the following multivariate merit threshold as specified by the experts in the Airbus group, namely the values



Objectives	Units	Role
flyover noise	dB	environmental impact
sideline noise	dB	environmental impact
nitrogen oxide emissions	lb	environmental impact
block fuel	lb	performance efficiency

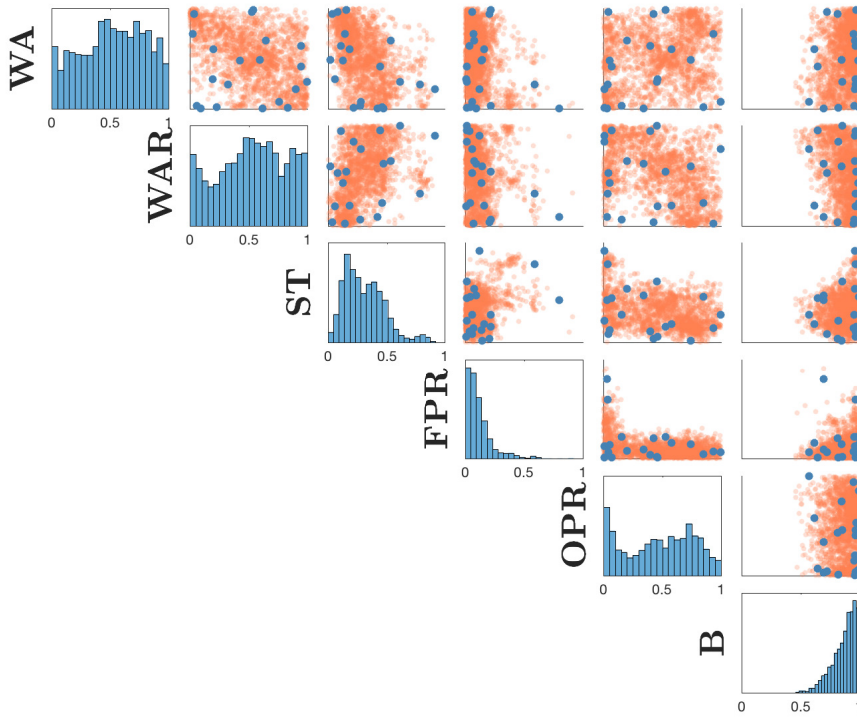
**Table 5.2:** *Output variables for the Airbus application.*

$$\mathbf{z} = (82, 86, 240, 33000)^\top, \quad (5.18)$$

where the vector  $\mathbf{z}$  is composed by the output variables shown in Table 5.2. The variability levels (the measurement error term in history matching) were elicited as

$$\boldsymbol{\sigma}_{\text{me}} = (0.1, 0.05, 5, 100)^\top, \quad (5.19)$$

where each component of  $\boldsymbol{\sigma}_{\text{me}}$  represent the corresponding variability bounds for each merit function. An LHS design was chosen to train a full Bayesian Gaussian process as described in Chapter 3. The number of training runs chosen was 60, as to follow the agreed upon rule of choosing 10 training points per input dimension in surrogate modelling [190]. The time to run the simulator model in Airbus' remote server, collect the data and distribute it among the participants was nearly 24 hours. Each run of the simulator code took approximately 1.5 hours. The use of Gaussian processes as surrogate models drastically improved each model evaluation. For this purpose an independent Gaussian process model was used for each simulator output with appropriately marginalised hyperparameters. This allowed to implement the history matching framework described in this chapter with the probabilistic version of the Second Maximum Implausibility function described by (5.16). Given the time constraints, it was only possible to perform one wave of refocussing for the history matching. However, the results shown in Figure 5.8 confirms a quick reduction of input parameter space. The orange dots show the NROY samples as found by the application of the TA<sup>2</sup>S<sup>2</sup> algorithm. The blue dots show the possible selection of additional 20 points in NROY space driven by the entropic learning criteria developed in Section 5.3.



**Figure 5.8:** *NROY identification for the Airbus aircraft design optimisation problem, solved through the analogy of a probabilistic history matching calibration problem. Orange dots correspond to the NROY samples using the  $TA^2S^2$  algorithm for the subset optimisation setting of probabilistic history matching. Blue dots depict potential candidates to further evaluate the simulator and refocus the Gaussian process emulators.*

## 5.7 Conclusion

In this chapter a full probabilistic approach to emulation has been taken into account for history matching. The Gaussian process model is used as a probabilistic emulator for simulators in history matching applications. In particular, it provides a probabilistic treatment of the implausibility function. The  $TA^2S^2$  algorithm developed in Chapter 4 have been used to sample uniformly from the NROY space. Additionally, a set of active learning functions have been proposed to guide how to select samples to improve the emulator. While they do not offer a consistent better alternative to that of uniform sampling, they provide a direction of future research in batch refocusing strategies.

The NROY space is identified through the use of the implausibility function.

Its formulation is analogous in spirit to the characterisation of failure domains in reliability analysis. This has led to study subset simulation as a sampling strategy used in reliability analysis. Subset simulation is formulated in a space of standard Gaussian variables where unitary scales are appropriate. However, if the intermediate failure domains become smaller and more complex in geometry, as in history matching, high rejection rates are encountered. This has led to work in Hamiltonian-based subset simulation presented in Chapter 6. The next chapter formulates both the reliability analysis problem and the subset simulation algorithm.

## 5.A Expected risk

The risk criterion is defined by taking into account both the target level  $z$  and the probability distribution for the output as learned from the emulator. Note that the output from a Gaussian process at index  $x$  is a Normal random variable with mean  $m(x)$  and variance  $\sigma^2(x)$ . In the following the reference to the index  $x$  is omitted to ease the exposition.

The risk is defined as a piecewise function, stemming from two possibilities. Firstly, as the shortage of reporting  $y$  units below the target level  $z$  when in expectation it should have reported a greater quantity. Secondly, when the report consisted of  $y$  units above the target level, when the expected value was known to be below the target. That is, the risk criterion can be written as

$$R_z(x) = \begin{cases} (\eta(x) - z)_+ & \text{if } m(x) \leq z \\ (z - \eta(x))_+ & \text{if } m(x) > z \end{cases}.$$

Taking into account that  $y$  is a random variable, the expected value of the risk is computed. Thus, the first component of the reported quantity can be written as

$$\begin{aligned} \mathbb{E}[R_z^-] &= \mathbb{E}[(z - y)_+] \\ &= \int_{-\infty}^z (z - y) f(y) dy \\ &= \sigma \left[ \left( \frac{z - m}{\sigma} \right) \Phi \left( \frac{z - m}{\sigma} \right) + \phi \left( \frac{z - m}{\sigma} \right) \right] \\ &= \sigma [\bar{z} \Phi(\bar{z}) + \phi(\bar{z})], \end{aligned}$$

where  $f(\cdot)$  denotes the density for the output of the simulator;  $(a)_+ = \max\{a, 0\}$ ;  $\bar{z}$  the standardised target level; and,  $\Phi$  and  $\phi$  the cumulative and density functions of a standard Gaussian random variable respectively. An analogous procedure let us write the second component of the risk function as

$$\begin{aligned} \mathbb{E}[R_z^+] &= \mathbb{E}[(y - z)_+] \\ &= \int_z^{\infty} (y - z) f(y) dy \\ &= \sigma [-\bar{z} \Phi(-\bar{z}) + \phi(\bar{z})]. \end{aligned}$$

Note that the first component is active whenever the inequality  $z - m < 0$  is

satisfied. Similarly, the second component becomes active when the inequality  $z - m > 0$  holds. These two expressions allow to write the expected risk learning criterion in compact form as

$$\mathbb{E}[R_z(a)] = \sigma(x) [-\text{sign}(\bar{z}) \bar{z} \Phi(-\text{sign}(\bar{z}) \bar{z}) + \phi(\bar{z})]. \quad (5.20)$$

## 5.B Entropic profile

Considering the GP prior for the computer model, the output of the simulator is a random variable. Thus, it makes sense to use the entropy of the output for a given interval on the prediction. Let's recall that Pukelsheim's rule is used to determine the implausibility of the output to identify the NROY space. Similarly, we can compute the entropic profile of the emulator response in the interval  $[z - k\sigma, z + k\sigma]$ , where  $k = 3$ . The entropic profile is then written as

$$\begin{aligned} H(y) &= \left| - \int_{z-k\sigma}^{z+k\sigma} \log f(y) f(y) dy \right| \\ &= \left| - \int_{z-k\sigma}^{z+k\sigma} \left[ -\frac{(y-m)^2}{2\sigma^2} - \log(\sqrt{2\pi}\sigma) \right] f(y) dy \right| \\ &= \left| \left[ \ln(\sqrt{2\pi}\sigma(x)) + 0.5 \right] [\Phi(\bar{z}_2) - \Phi(\bar{z}_1)] - 0.5 [\bar{z}_2\phi(\bar{z}_2) - \bar{z}_1\phi(\bar{z}_1)] \right|, \end{aligned}$$

where  $\bar{z}_1$  and  $\bar{z}_2$  denote the standardised contour levels. That is,  $\bar{z}_1 = (z - m - k\sigma)/\sigma$  and  $\bar{z}_2 = (z - m + k\sigma)/\sigma$ . The last inequality is attained after applying some properties of the integrals of standard Gaussian densities.

## 5.C Random functions from a Gaussian process prior

As there are no standard test functions for history matching in increasing dimensional settings, it is proposed to generate random test cases from a Gaussian process prior. This is a similar strategy followed in [115] where in different dimensional settings functions are generated randomly. This is achieved by using the posterior mean of a Gaussian process as such a random function.

The process is summarised as follows. For each dimensional setting  $d$ ,  $n = 100 \times d$

points are generated uniformly at random from  $[0, 1]^d$ . Let us denote these chosen seeds as  $\mathbf{X} \in \mathbb{R}^{n \times d}$ . The lengthscales of a Matern kernel,  $\boldsymbol{\varphi}$ , are generated from a uniform random vector in the cube  $[0, 2]^d$  and the signal noise is chosen as 10. The random choice of seeds and lengthscales generate different functions during this process. The choice of the signal noise to be the same for every test case allows one to make comparisons in terms of predicted error and CRPS among the functions within the same dimensional setting.

The evaluation of the random function is performed as follows. The Gaussian process prior defines a multivariate Gaussian distribution for the output on the seeds  $\mathbf{f} \sim \mathcal{N}(0, 10^2 \mathbf{K})$ , where  $\mathbf{K}$  denotes the covariance matrix using the Matern kernel, the seeds  $\mathbf{X}$  and lengthscales chosen as above. Thus, given a set of training input configurations, say  $\mathbf{X}'$ , the output of the random function is

$$\mathbf{y} = \mathbf{k}_*^\top \mathbf{K}^{-1} \mathbf{f}$$

where  $\mathbf{k}_*$  is the column vector of pairwise evaluations of the chosen kernel between each training run  $\mathbf{X}'_i$  and all random seeds  $\mathbf{X}$ .

---

### Reliability Analysis and Hamiltonian Dynamics<sup>1</sup>

---

This chapter studies the problem of reliability analysis in engineering systems. As discussed in Chapter 1 the reliability analysis problem can be formulated by a performance function which represent the state of a complex system. Thus, the reliability of a system is studied through the characterisation of a failure domain. In particular, the failure domain is defined as the response of the system exceeding a given safety threshold. The main objective is to estimate the probability of failure of a system, that is, the probability of the demand exceeding the system's capacity. This formulation can be used to draw an analogy to the history matching problem presented in Chapter 5. In the context of reliability analysis, the implausibility is the performance function, and the number of standard deviations is the safety threshold level. The observation that in history matching the identification of the NROY space can be thought as identifying a contour level of the simulator within some variability bands, has motivated the question of efficient sampling in highly constrained spaces. On the other hand, the subset simulation algorithm (to be discussed in more detail in this chapter) has been the go-to-approach to simulate and estimate the probability of failure in reliability analysis. In particular, subset simulation has proven to be a successful approach to quantifying such

---

<sup>1</sup>The results and ideas have been submitted for publication as a manuscript, see Garbuno-Inigo et al. [87]

negligible probabilities with much lower computation cost compared to typical Monte Carlo estimates. Nonetheless, it is known that subset simulation is prone to high rejection rates when the failure domain is highly constrained, and this has motivated the development of several adaptive techniques [12]. In light, of the results of incorporating adaptive scaling and delayed rejection in Chapter 4, this chapter seeks to improve the efficiency of subset simulation to be used in reliability analysis in the context of engineering computations. In particular, Hamiltonian Monte Carlo is proposed in the context of subset simulation as it uses first order information to guide the generation of samples.

## 6.1 Introduction

Subset simulation (SuS) [9, 12] has proven to be an efficient rare event simulation method to estimate the probability of failure in complex systems. It has been particularly successful in high dimensional settings. The idea behind the algorithm is to generate a sequence of nested intermediate failure subdomains that keep both the complexity of the region of interest and the number of simulations as low as possible.

In summary, the reliability problem is modelled by an input vector of random variables  $\mathbf{x} \in \mathbb{R}^d$  and a system performance function  $h(\mathbf{x})$ . Randomness in  $\mathbf{x}$  allows one to characterise uncertain behaviour on the system. A prespecified threshold level  $b$  determines its correct state. Without loss of generality, failure is achieved whenever the system response exceeds the threshold,  $h(\mathbf{x}) > b$ .

In applications it is common to assume independent standard Gaussian variables  $\mathbf{u}$ . For this purpose, a mapping  $\mathbf{u} = T(\mathbf{x})$  is needed. Typical choices are the Rosenblatt [116] or the Nataf [60] transformations. The information available to the analyst regarding the random variables is what determines which transformation to use. To be explicit, if the joint distribution or marginal information are known to the analyst. Without loss of generality, this chapter assumes a standard Gaussian space for the random variables.

High dimensional applications of Subset Simulation have been successful as the algorithm is both amenable to parallelisation and is efficient to generate random vectors. Subset Simulation applies localised Markov chain Monte Carlo, which traditionally uses component-wise updates [127, 227, 178]. However, it is known



that towards the final levels the rejection rates of samples increases substantially. Thus, the exploration of the input domain is less efficient and scaling the width of the proposals is needed to retain movement along a chain. The reason behind this is that the generation of samples for Subset Simulation is usually seen as a two step acceptance procedure. The first step generates a good proposal according to the standard Gaussian space. The second step checks if the proposed sample lies within the intermediate failure domain. It can be seen that not including the failure domain information potentially increases the rejection of samples. For example, consider a single random variable  $u$  and a threshold  $b$ . Let us suppose that the current state of the Markov chain is in the threshold. A Gaussian proposal will include values below the threshold. The two step acceptance will potentially discard half of the proposals. As the dimensionality increases, this can be seen to become cumbersome and would create a negative impact on the efficiency of the method.

In this chapter, Hamiltonian Monte Carlo is proposed to be used to generate samples for the Markov chains needed for Subset Simulation. The failure domain is incorporated in the proposals by means of a smooth barrier function. As it will be seen, the smoothness of the barrier alleviates the non-differentiability issues of incorporating the second acceptance step in the generation of samples. Moreover, this formulation of the method allows an easy implementation of the algorithm in general purpose probabilistic programming software such as Stan [40] or PyMC3 [192]. Overall, the Hamiltonian-based Subset Simulation achieves competitive performance on typical tasks such the linear response system. It is also shown to perform well on highly constrained failure domains and stochastic finite element applications.

The chapter is structured as follows. A formal definition of both the reliability analysis problem and the Subset Simulation algorithm is presented in Section 6.2. A brief overview on Hamiltonian Monte Carlo and the details of Hamiltonian-based Subset simulation is found in Section 6.3. Numerical experiments are presented in Section 6.4. Lastly, concluding remarks are presented in Section 6.5.

## 6.2 Reliability Analysis and Subset Simulation

Subset Simulation (SuS) [9, 12] is an advanced Markov chain Monte Carlo method for reliability and failure analysis of complex systems, especially designed for simulating rare events. It is based on the idea that a small failure probability can be expressed as a product of larger conditional failure probabilities. This effectively converts a rare simulation problem into a series of more frequent ones.

### 6.2.1 Reliability Analysis

A failure event can be formulated as the exceedance of a critical response  $Y \in \mathbb{R}$  over a prescribed threshold  $b$ . Let  $Y = h(\mathbf{u})$  be the response quantity of interest depending on random parameters  $\mathbf{u} \in \mathbb{R}^d$  distributed according to the probability density function (PDF)  $\pi(\mathbf{u})$ . The *performance function*  $h(\cdot)$  represents the relationship between the uncertain input parameters and the output response. It contains all the available information of the system such as stresses, loads, etc. The primary interest of reliability analysis is to determine the *failure probability*  $P(Y > b)$ :

$$P_F = P(Y > b) = \int \pi(\mathbf{u}) \mathbf{1}(\mathbf{u} \in F) d\mathbf{u}, \quad (6.1)$$

where

$$F = \{Y > b\} = \{\mathbf{u} \in \mathbb{R}^d : h(\mathbf{u}) > b\}, \quad (6.2)$$

denotes the failure event or the failure region in the parameter space, depending on the context.  $\mathbf{1}(\cdot)$  is an indicator function, equal to 1 if its argument is true and zero otherwise.

SuS approximates the solution to Equation (6.1) by generating a sequence of random samples of  $\mathbf{u}$  conditional on increasingly rare failure events  $F = F_M \subseteq \dots \subseteq F_2 \subseteq F_1$ . Each intermediate failure event in the sequence is define as

$$F_i = \{Y > b_i\} \quad (6.3)$$

where  $\{b_i\}_{i=1}^M$  is an increasing sequence of threshold values adaptively determined during the simulation run, and  $M$  is the final intermediate level. The probability of failure is approximated by exploiting the conditional dependence on intermediate

failure levels. That is, by writing the target failure probability as

$$P_F = P\{Y > b\} = P(F_M|F_{M-1}) \cdots P(F_1|F_0), \quad (6.4)$$

where  $F_0$  denotes the unconditional event.

The samples are used to estimate the complementary cumulative distribution function (CCDF) of  $Y$ . When the right tail covers the threshold value associated with the target failure event, the required failure probability can be obtained from the estimate of the CCDF. The conditional samples can also be used for estimating the conditional expectation in probabilistic failure analysis, a feature not shared by conventional variance reduction techniques.

### 6.2.2 Subset Simulation

Subset simulation requires two input parameters: the number of samples generated at each failure level, denoted by  $N$ , and the conditional level probability, denoted by  $p_0$ . For algorithmic convenience, both are chosen such that  $p_0 \times N$  and  $1/p_0$  are positive integers. It has been previously studied in [227] that a prudent choice for the conditional probability satisfies  $p_0 \in [0.1, 0.3]$ .

A simulation run starts with the (unconditional) level 0, where  $N$  independent and identically distributed (*iid*) samples of  $\mathbf{u}$  are generated from  $\pi(\cdot)$ , *i.e.* by direct Monte Carlo. The corresponding values of  $Y$  are computed and sorted in ascending order, resulting in the ordered list  $\{b_k^{(0)} : k = 1, \dots, N\}$ . The value  $b_k^{(0)}$  gives the estimate of  $b$  corresponding to the exceedance probability  $p_k^{(0)} = P(Y > b)$  where

$$p_k^{(0)} = \frac{N - k}{N}, \quad k = 1, \dots, N. \quad (6.5)$$

The first failure threshold level is defined by means of the  $(p_0N + 1)$ -th largest sample value of  $Y$ . Thus, the conditional failure relation

$$p_0 = P\{Y > b_1\} = P(F_1|F_0), \quad (6.6)$$

is satisfied. Note that the  $p_0N$  top ranked samples have responses greater or equal to  $b_1$ . By construction, these samples are already inside the intermediate failure level  $F_1$ . The generation of new samples from the intermediate failure domain is done by exploiting this property. The  $p_0N$  top ranked samples are used as seeds to grow independent Markov chains from the target density  $\pi(\mathbf{u}|F_1) \propto$

$\pi(\mathbf{u})\mathbf{1}(\mathbf{u} \in F_1)$ . This results in growing  $N_c = p_0 N$  chains, each with length  $N_s = N/N_c = 1/p_0$ . Having the seeds inside the target region,  $F_1$ , allows one to discard any burn in period usually required in MCMC simulations to grow a single Markov chain. The SuS algorithm follows the previous principle in an iterative manner. The  $i$ -th level (for  $i \geq 1$ ) is conditional on the intermediate failure event  $\{Y > b_i\}$ , where  $b_i$  is determined as the  $(N_c + 1)$ -th largest sample value of  $Y$  from level  $i - 1$ . Thus, at each intermediate failure level  $i$  the relation  $p_0 = P(F_i|F_{i-1})$  is satisfied. At level  $i$ ,  $N_c$  independent Markov chains are grown from the target density  $\pi(\cdot|F_i)$ , each with length  $N_s$ .

The process above is repeated until the target threshold level is reached. As before, let us denote by  $M$  the final intermediate level. The threshold value satisfies  $b_M \geq b$  and thus the number of conditional samples with responses greater than  $b$ , exceeds  $N_c$ . The estimate of the failure probability is derived from equation (6.4), which can be written as

$$\hat{P}_F = p_0^{M-1} \frac{1}{N} \sum_{k=1}^N \mathbf{1}(\mathbf{u}_k \in F_M), \quad (6.7)$$

where  $\frac{1}{N} \sum_{k=1}^N \mathbf{1}(\mathbf{u}_k \in F_M)$  is the estimate of the conditional failure probability at level  $M$ , say  $\hat{P}_M$ .

### 6.2.3 MCMC Proposals for Subset Simulation

SuS relies on the generation of samples at each annealing level. Thus, the efficiency of the SuS algorithm depends on the efficiency of generating samples at each conditional level. Recall that at each intermediate, say the  $i$ -th,  $N_c$  chains are grown independently, starting from a given seed in  $F_i$ . The chain is grown using a transition proposal. Let us denote by  $\mathbf{u}$  the current state of the chain and  $\mathbf{u}^*$  the proposed candidate of the transition. Let us denote by  $q(\mathbf{u}^*|\mathbf{u})$  the transition density. An ideal property of a Markov chain is to be *ergodic* with respect to the transition operator [92]. To enable this, the proposals for new states of the Markov chain are designed to satisfy the detailed-balance condition. This means that new samples are generated so that the following equation is satisfied

$$p(\mathbf{u}|\mathbf{u}^*) \pi(\mathbf{u}^*|F_i) = p(\mathbf{u}^*|\mathbf{u}) \pi(\mathbf{u}|F_i). \quad (6.8)$$

Detailed-balance further implies that the proposal distribution leaves the target distribution invariant. That is, that the transition operator preserve the target distribution  $\pi(\cdot|F_i)$  for the proposed state. This can be written as

$$\pi(\mathbf{u}^*|F_i) = \int p(\mathbf{u}^*|\mathbf{u}) \pi(\mathbf{u}|F_i) d\mathbf{u}, \quad (6.9)$$

where both sides of equation (6.8) are integrated with respect to  $\mathbf{u}$ .

In light of the above, several variants of SuS have been proposed using different transition operators  $p(\mathbf{u}^*|\mathbf{u})$ . The simplest algorithm is the Metropolis Hastings (MH) operator in which the acceptance of a new state for the chain is accepted with probability

$$a(\mathbf{u}, \mathbf{u}^*) = \min \left\{ 1, \frac{p(\mathbf{u}|\mathbf{u}^*) \pi(\mathbf{u}^*|F_i)}{p(\mathbf{u}^*|\mathbf{u}) \pi(\mathbf{u}|F_i)} \right\}. \quad (6.10)$$

The acceptance probability preserves the detailed-balance condition in equation (6.8). Note that the current state of the Markov chain already satisfies  $\mathbf{1}(\mathbf{u} \in F_i)$ . This observation means that  $\mathbf{1}(\mathbf{u} \in F_i) = 1$ , which simplifies the Metropolis Hastings ratio in (6.10) to

$$a(\mathbf{u}, \mathbf{u}^*) = \min \left\{ 1, \frac{p(\mathbf{u}|\mathbf{u}^*) \pi(\mathbf{u}^*)}{p(\mathbf{u}^*|\mathbf{u}) \pi(\mathbf{u})} \right\} \mathbf{1}(\mathbf{u}^* \in F_i), \quad (6.11)$$

$$= \hat{a}(\mathbf{u}, \mathbf{u}^*) \mathbf{1}(\mathbf{u}^* \in F_i), \quad (6.12)$$

where the overall acceptance probability of transitioning,  $a(\mathbf{u}, \mathbf{u}^*)$ , can be seen as a two stage proposal. The first stage is a proposal transition from the initial target distribution  $\pi(\cdot)$ . The second, checks if the proposed candidate lies within the intermediate failure region  $F_i$ . In the remainder, the first stage will be referred to as the *proposal step*; the second, the *failure step*. This view has influenced the development of variants for the transition operator which aim to improve upon the efficiency of the MH algorithm applied to SuS. For example Katafygiotis and Zuev [127] applied a geometric study to understand the difference in performance between the MH algorithm and the original sampling strategy of SuS, the modified MH (MMH) [9]. The latter being a composition of one dimensional independent proposals in the components of vector  $\mathbf{u}$ . Once the proposal is generated, it is later evaluated in the failure step. Overall, the improved efficiency of MMH lies in the ability to generate samples from within the typical set of the target  $\pi(\cdot)$  distribution.

The typical set can be thought of as the region of the support of a distribution where it concentrates most of its mass. For example, a  $d$ -dimensional multivariate Gaussian concentrates its mass on the shell of a sphere with radius  $\sqrt{d}$ . As the dimension increases the width of the shell decreases. Other approaches include an infinite limit of proposals which is applied component-wise to the random vector (iMH) [11]. Spherical subset simulation [125] is another variant of the SuS method which exploits the typical set approach of the density  $\pi(\cdot)$ . It decomposes the failure region as intersections of the failure region with typical sets on Gaussian space with an increasing sequence of radii. Other approaches can be found in [178, 177, 227, 36]. It is worth mentioning that by means of component-wise updates SuS is capable to scale to high dimensional spaces typical in the reliability analysis literature.

## 6.3 Hamiltonian Monte Carlo

As mentioned in Section 6.2.3, most sampling algorithms developed for SuS rely on the two step factorization of the acceptance probability (6.12). As effective as they are for proposing new candidates for the target  $\pi(\cdot)$ , the rejection rates on the failure step grows larger as the intermediate failure domains become smaller. In particular, the failure region could potentially not be included within the typical set, which makes the generation of samples by the two step proposal inefficient.

Hamiltonian Monte Carlo (HMC), originally called Hybrid Monte Carlo [69], is an auxiliary variable sampling algorithm. It uses an additional random vector  $\mathbf{v} \in \mathbb{R}^d$  and aims to sample from the joint density function

$$\pi(\mathbf{u}, \mathbf{v}) = \pi(\mathbf{v}|\mathbf{u}) \pi(\mathbf{u}), \quad (6.13)$$

where the marginal with respect to the auxiliary variable follows the target distribution  $\pi(\cdot)$ . HMC defines the joint distribution by means of the energy function

$$\begin{aligned} H(\mathbf{u}, \mathbf{v}) &\equiv -\log \pi(\mathbf{u}, \mathbf{v}) \\ &= -\log \pi(\mathbf{v}|\mathbf{u}) - \log \pi(\mathbf{u}) \\ &\equiv K(\mathbf{v}, \mathbf{u}) + V(\mathbf{u}), \end{aligned} \quad (6.14)$$

where  $K(\cdot)$  and  $V(\cdot)$  are interpreted as the kinetic and potential energies of a Hamiltonian system. HMC alternates between generating random momentum variables  $\mathbf{v}$  and simulating the Hamiltonian dynamics of the system by solving

$$\frac{d\mathbf{u}}{dt} = \frac{\partial H}{\partial \mathbf{v}}, \quad \frac{d\mathbf{v}}{dt} = -\frac{\partial H}{\partial \mathbf{u}}, \quad (6.15)$$

which generates both  $\mathbf{u}^*$  and  $\mathbf{v}^*$ , referred as position and momentum random vectors respectively. This method has been revisited in [202] in the context of Information Theory and Itô stochastic differential equations. The new state of the extended system is proposed as  $\mathbf{u}^*$  with reversed momentum  $-\mathbf{v}^*$ . This reversal is acknowledged to preserve the reversibility of the Markov chain. The joint model (6.13) allows one to discard the momentum variables and retain the target marginal distribution as the original target of the problem. A common choice for the Kinetic energy function is a squared model  $K(\mathbf{v}) = 0.5 \mathbf{v}^\top \Sigma^{-1} \mathbf{v}$ . This choice corresponds to an independent Gaussian model for the momentum. The variance-covariance matrix  $\Sigma$  is chosen as a pre-conditioner if a known variance-covariance structure is known for the position variables [29, 166]. If there is an explicit dependence between position and momentum, this can be modelled by defining  $\Sigma(\mathbf{u})$ . This version is known as Riemannian Manifold Hamiltonian Monte Carlo [98].

The solution of the Hamiltonian system (6.15) usually does not have an analytic expression. Thus, the most common approach is to approximate the solution by means of a symplectic integrator. In the sampling context, these integrators preserve the detailed-balance condition of the proposal distribution. Thus, the use of symplectic integrators preserves the target distribution of the states of the Markov chain. The most common choice is the leap-frog integrator which sequentially repeats

$$\begin{aligned} \mathbf{v}_{t+\frac{\epsilon}{2}} &= \mathbf{v}_t - \frac{\epsilon}{2} \frac{\partial V}{\partial \mathbf{u}} \Big|_t, \\ \mathbf{u}_{t+\epsilon} &= \mathbf{u}_t + \epsilon \frac{\partial K}{\partial \mathbf{v}} \Big|_{t+\frac{\epsilon}{2}}, \\ \mathbf{v}_{t+\epsilon} &= \mathbf{v}_{t+\frac{\epsilon}{2}} - \frac{\epsilon}{2} \frac{\partial U}{\partial \mathbf{u}} \Big|_{t+\frac{\epsilon}{2}}, \end{aligned} \quad (6.16)$$

where the time variable  $t$  is referred to as the integration time, and  $\epsilon$  as the stepsize. Both parameters need to be defined to perform the Hamiltonian dynamics. The efficiency of the sampler is strongly connected to an appropriate selection of

these. A short integration time or a very small stepsize leads to random walk behaviour. This causes the Markov chain to evolve slowly, thus having highly correlated samples which do not explore the support efficiently. On the other hand, a poorly chosen stepsize leads to inaccurate simulation of the Hamiltonian dynamics, thus introducing numerical errors in the proposal for the Markov chain. The MH ratio is used to cope with numerical errors in the simulation of the trajectory. This stabilises the numerical procedure of simulating the Hamiltonian dynamics. Appropriate choices of these essential parameters have been studied in the literature and the state-of-the-art is the non-u-turn (NUTS) variant [114]. A general purpose implementation of the latter can be found in Stan, a probabilistic programming language for statistical inference [40].

### 6.3.1 HMC for Subset Simulation

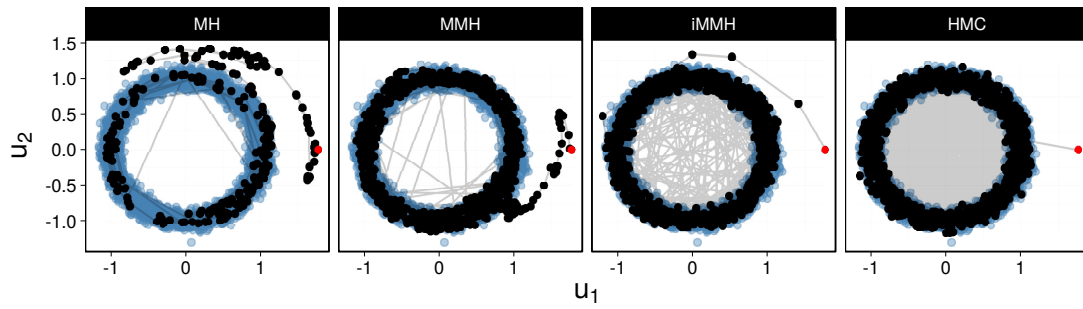
HMC has proven to be efficient in high-dimensional problems. Its ability to generate samples by navigating the joint probability density makes it an appealing candidate to be applied in subset simulation problems. The main drawback is that HMC uses gradient information from both the Kinetic and Potential functions, as seen in (6.16). The Kinetic function  $K(\cdot)$  usually does not represent any problem, since it is chosen conveniently from a Gaussian model, namely,  $K(\mathbf{v}) = 0.5 \mathbf{v}^\top \Sigma^{-1} \mathbf{v}$ . However, the target density in SuS applications incorporates an indicator function for which the gradient is not easily available. The indicator functions found in the conditional densities (6.4) can be reinterpreted as a constrained sampling problem. This has been studied previously in the statistics literature as mentioned in the review chapter of Neal [166]. In the latter, the proposed solutions are to either incorporate a reflection step when the particle hits the constraints barrier or add a penalising energy function that retains the dynamics inside the constraints. Adding a reflection step incorporates a numerical procedure that involves estimating the time for the particle to hit the barrier. The penalising energy function adds a parameter which is highly sensitive to small changes and difficult to elicit. The former approach has been studied further in [157] leading to a leap-frog integrator that incorporates a reflection/refraction component. A similar approach was followed in [220] in which the time component is estimated through first order Taylor approximations and specifically developed for SuS. An alternative approach is followed in this chapter which is inspired by considering



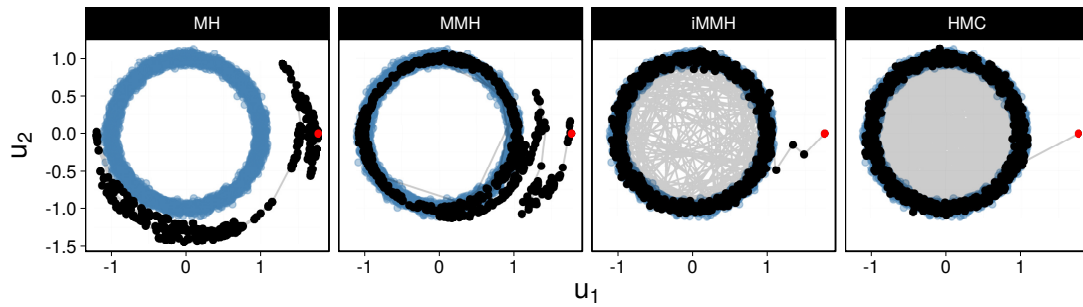
typical sets and approximation methods found in interior point optimisation algorithms.

The discussion in Section 6.2.3 suggests the idea of alternating between a proposal step and a failure step as done in the MH, MMH and iMMH variants of subset simulation. Let us assume the current seed of a Markov chain is found far off the typical set. This could be the case for systems with a linear performance function  $\sum_{j=1}^d \mathbf{u}_j$  and a high threshold value, or by some performance function that induces failure when  $\|\mathbf{u}\|^2$  is greater than a given threshold. All previously discussed SuS variants aim to maintain the ergodicity of the proposals with respect to  $\pi(\cdot)$ , which without loss of generality can be assumed to be a  $d$ -dimensional Gaussian. Applying a naive HMC proposal step will aim at targeting the ergodic distribution. Projections of high dimensional vectors are shown in Figure 6.1 in order to compare the effect of the Markov chain proposals under MH, MMH, iMMH, and HMC. This projection is achieved by normalising the complete vector by its Euclidean norm and transforming the first two components of the normalised vector to polar coordinates. Blue dots in Figure 6.1 show samples drawn from the true stationary distribution. That is, the corresponding sample vectors are generated by drawing  $d$  independent Gaussians. This shows how the width of the typical set decreases as the dimensionality increases and the samples get concentrated in the outer shell. The red dot shows the starting point of the Markov chain. The starting point is chosen to be outside of the typical set. Black dots are samples generated by different variants of SuS. The lines connecting the samples show the transitions between two consecutive states of the Markov chain. All samplers are used with optimal scaling parameters to increase the efficiency of the chain, [see 88, 127, 11, 40, for more details]. Note how the the MH and MMH struggle to move towards the typical set. However, the local movements of the transitions illustrate the efficiency that has been observed in the literature for the MMH in high-dimensional settings. The iMMH variant rapidly reaches the typical set, with a good mixture between local and aggressive transitions. The aggressive transitions are shown as movements to the opposite side of the ring. For this example, HMC achieves the highest efficiency as it both reaches the typical set without effort and maintains quick transitioning along the ring. The exercise was carried out for 100 and 300 dimensions as denoted in Figure 6.1.

Although encouraging, these results should be taken with caution. The local movements of MMH and iMMH confirm their efficiency in the SuS context. Recall



(a) Projection of typical set of a  $d$ -dimensional Gaussian,  $d = 100$



(b) Projection of typical set of a  $d$ -dimensional Gaussian,  $d = 300$

**Figure 6.1:** Typical set and Markov chain sampling for  $d$ -dimensional Gaussian random vectors. Blue dots denote the typical set of the corresponding multivariate standard Gaussian. The red dot denotes the starting point of the Markov chain. The black dots depict the samples generated by each variant of MCMC sampling. The lines connecting the dots show the transitions of the Markov chain.

that for SuS the growth of Markov chains is done by alternating between a proposal and failure step as described by (6.12). This means that the proposal step is just a one step Markov chain that will aim to move towards the typical set with respect to  $\pi(\cdot)$ , the ergodic distribution. Applying a proposal step with HMC seems to be too aggressive, since it rapidly returns to the typical set. If the typical set of  $\pi(\cdot|F_i)$  is substantially different from that of  $\pi(\cdot)$ , this will lead to high rejection rates in the failure step. It is clear that to be able to harness all the benefits from HMC in the SuS context, the two stage approach to sample from intermediate failure domains should be avoided.

### 6.3.2 Smooth failure domains

As mentioned in Section 6.2.2, SuS aims to sample from nested intermediate failure domains  $F_i$ . The target density being proportional to the product  $\pi(\mathbf{u}) \mathbf{1}(\mathbf{u} \in F_i)$ , where  $\mathbf{1}(\cdot)$  is the indicator function for the argument. Moreover, each intermediate failure region is defined as

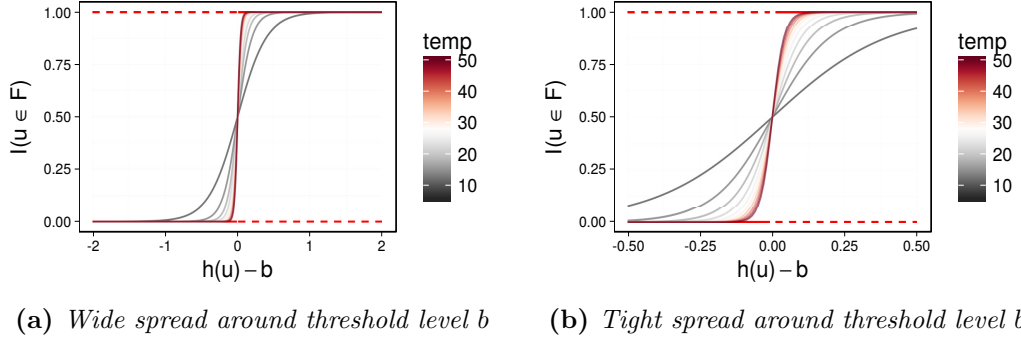
$$F_i = \{\mathbf{u} \in \mathbb{R}^d : h(\mathbf{u}) - b_i > 0\}, \quad (6.17)$$

where  $h(\cdot)$  denotes the system performance function. The indicator function  $\mathbf{1}(\cdot)$  is incorporated to retain samples within the failure region  $F_i$  and it defines the conditional distribution  $\pi(\cdot|F_i) \propto \pi(\cdot) \mathbf{1}(\cdot \in F_i)$ . However, this prevents  $\pi(\cdot|F_i)$  to be a differentiable function to be used in simulation of the Hamiltonian system (6.16). In the same spirit as in interior point methods in optimisation, the indicator function is approximated by a smooth surrogate. Since the output of the indicator is defined by the sign of  $h(\mathbf{u}) - b_i$ , a sigmoid is proposed as a viable smoothing function. A *temperature* parameter can be incorporated in the sigmoid to control how close this surrogate is to the true indicator function. The sigmoid with temperature  $t$  has expression

$$s_t(\mathbf{u}) = \frac{1}{1 + \exp(-t g_i(\mathbf{u}))}, \quad (6.18)$$

where  $g_i(\mathbf{u}) = h(\mathbf{u}) - b_i$ . In Figure 6.2 a sequence of sigmoid approximations is shown with increasing values of the temperature parameter.

It is important to note that as the temperature increases the sigmoid becomes sharper around the origin. The sharp behaviour near the origin helps to decide whether a given input vector  $\mathbf{u}$  belongs to the intermediate failure level  $i$ . The magnitude of the temperature is also associated as to how concentrated the output of the performance function  $h(\cdot)$  is around the threshold level  $b_i$ . If the range of values presents a high spread, as shown in Figure 6.2a, moderately low values would suffice to make the distinction. On the other hand, if the the output is tightly concentrated around the threshold value, a larger temperature would be needed to make the sigmoid sharper around the origin, see Figure 6.2b. Nonetheless, the subset simulation algorithm allows a framework to adaptively learn an appropriate temperature value. The samples at level  $i - 1$  can be used to make such a choice. Under the Gaussian assumptions for the joint variables  $(\mathbf{u}, \mathbf{v})$ , and using the



**Figure 6.2:** Approximation of an indicator function by means of a sigmoid with an increasing sequence of temperature values. Note that in the left panel the sigmoid's sharpness seems appropriate for values in the  $[-2, 2]$  neighbourhood. In contrast, the right panel shows that a higher temperature would be needed if the simulator output does not exhibit a high variation. This is referred in the text as a tight neighbourhood for the simulator output.

sigmoid function (6.18), the total Hamiltonian energy (6.14) can be written with the kinetic and potential functions

$$K(\mathbf{v}) = -\log(\pi(\mathbf{v})), \quad (6.19)$$

$$V(\mathbf{u}) = -\log(\pi(\mathbf{u})) + \log(1 + \exp(-t g_i(\mathbf{u}))), \quad (6.20)$$

respectively, where  $\pi(\cdot)$  is the kernel of a  $d$ -dimensional Gaussian. Simulating the Hamiltonian requires the explicit partial derivatives of the energy functions. From the above equations is clear that the kinetic function does not change with respect to the original formulation. The interest is focused in the new potential function, which induces a change in the updates of the momentum variables

$$\begin{aligned} \frac{d\mathbf{v}}{dt} &= -\frac{\partial V}{\partial \mathbf{u}} \\ &= \frac{t}{1 + \exp(t g_i(\mathbf{u}))} \nabla h(\mathbf{u}) - \frac{1}{\pi(\mathbf{u})} \nabla \pi(\mathbf{u}), \end{aligned} \quad (6.21)$$

where  $\nabla h(\cdot)$  denotes the gradient of the performance function. The expression for the density of the target variable  $\mathbf{u}$  has been left to accommodate for more general settings. However, under a standard Gaussian model, the gradient of the log-density, the second term in (6.21), is equal to  $\mathbf{u}$ . Note that when the sample  $\mathbf{u}$  satisfies  $g_i(\mathbf{u}) > 0$ , specially for large values, the first term could effectively be zero.

Thus, the first term in (6.21) dampens the exploration beyond the intermediate failure domains. This effectively leads the particles towards the regions of interest.

## 6.4 Numerical experiments

The main advantage of the formulation with the sigmoid function (6.18) is that it is widely applicable in probabilistic programming interfaces. Experiments in Sections 6.4.1 and 6.4.3 were easily translated into the Stan probabilistic programming language [40] as it is possible to incorporate terms such as (6.18) into the target density. Unless otherwise stated, the numerical experiments in this section were performed using Stan. Other probabilistic programming alternatives are PyMC3 [192] or Edward [213].

For the following experiments, the coefficient of variation (cov) of the samples is reported as a measure of the efficiency of the algorithm. The probability of failure for each simulation is also reported. The cov is calculated according to the derivation in [9], which is calculated as

$$\delta^2 = \sum_{i=1}^M \delta_j^2, \quad (6.22)$$

where  $\delta$  is the coefficient of variation of the  $\hat{P}_F$ , and  $\delta_j$  is the coefficient of variation of the estimate of the conditional probabilities  $\hat{P}_j = \hat{P}(F_j|F_{j-1})$ . Each  $\delta_j$  can be estimated by

$$\delta_j^2 = \frac{1 - p_0}{p_0} \left( \frac{1 + \gamma_j}{N} \right), \quad (6.23)$$

where

$$\gamma_j = 2 \sum_{k=1}^{N_s-1} \left( 1 - \frac{k}{N_s} \right) \rho_j(k), \quad (6.24)$$

where  $N_s$  is the number of samples generated at every chain and  $\rho_j(k)$  is the  $k$ -lag autocorrelation coefficient of the indicators on the failure sets. More explicitly, let us denote by  $\mathbf{u}_{i,r}^{(j-1)}$  the  $r$ -th sample of the  $i$ -th Markov chain grown at failure level  $j - 1$ . Let  $Y_{i,r}^{(j-1)} = h(\mathbf{u}_{i,r}^{(j-1)})$  denote the response of the sample, and recall that samples generated at level  $j - 1$  are used for the computation of the intermediate

failure level  $b_j$ . The expression to compute the  $k$ -lag autocorrelation is written as

$$\rho_j(k) = \frac{1}{p_0(1-p_0)} \left[ \frac{1}{N_c(N_s-k)} \left[ \sum_{i=1}^{N_c} \sum_{r=1}^{N_s-k} I(Y_{i,r}^{(j-1)} > b_j) I(Y_{i,r+k}^{(j-1)} > b_j) \right] - p_0^2 \right]. \quad (6.25)$$

All experiments were run under typical configurations for subset simulation. That is, the level probability is chosen as  $p_0 = 0.1$  and  $N = 1000$  [in 228, the choice of  $p_0$  is investigated]. The number of chains and samples generated within the chain are set to  $N_c = N \times p_0$  and  $N_s = N/N_c$  respectively. Any departure from these values is explicitly stated.

### 6.4.1 Linear performance function

In this example a linear performance function is considered for different dimensional settings. Assuming the random vector  $\mathbf{u} \in \mathbb{R}^d$  is distributed as a standard multivariate Gaussian, the performance function is defined as

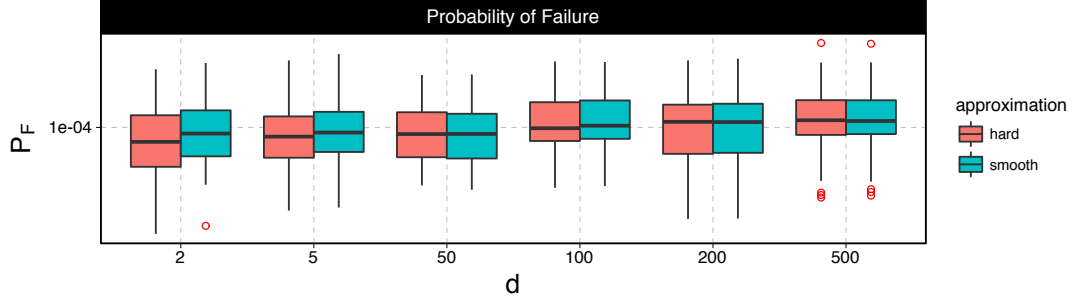
$$h(\mathbf{u}) = \sum_{l=1}^d u_l, \quad (6.26)$$

where it follows by independence of the components that  $h(\mathbf{u}) \sim \mathcal{N}(0, d)$ . Thus, the probability that the performance function exceeds a given threshold can be computed as

$$P\{h(\mathbf{u}) > b\} = 1 - \Phi\left(\frac{b}{\sqrt{d}}\right), \quad (6.27)$$

where  $\Phi(\cdot)$  denotes the standard Gaussian cumulative function. Keeping the true probability of failure fixed at  $10^{-4}$ , the value of the threshold is computed automatically for every dimensional setting using (6.27). For every dimensional setting  $d \in \{2, 5, 50, 100, 200, 500\}$ , 50 independent runs of the Hamiltonian-based subset simulation were performed. The temperature of the sigmoid function was chosen as  $t = 15$  as an initial Monte Carlo estimate of the variance of the response function was not highly concentrated as previously discussed in Section 6.3.2. The results are summarised as boxplots for each dimensional setting. As shown in Figure 6.3, it can be noted that the probability estimates agree on average with the true probability of failure, regardless of dimensionality. Moreover the

estimates are contained within an order of magnitude. Two different estimation methods are reported for comparison. The *hard* estimator uses the indicator function to indicate the sample is a member of the failure domain. The *smooth* estimator uses the sigmoid function with the temperature provided. Figure 6.3 shows evidence that both estimators agree, since in each pair of boxplots, there is no major discrepancy.

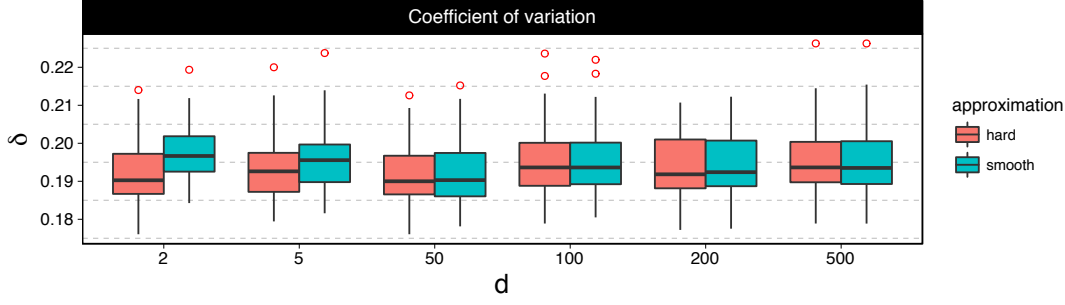


**Figure 6.3:** Results for the linear performance function in different dimensional settings  $d$ . The *hard* approximation uses the indicator functions for the computation of the probability of failure in (6.7). Whereas the *smooth* approximation refers to using the sigmoid (6.18) instead for such approximation.

Figure 6.4 depicts the total coefficient of variation for the linear response functions for different dimensional settings. It can be noted that the coefficient of variation exhibited by the Hamiltonian-based Subset Simulation is  $\delta = 0.3$ , which agrees with what was reported in [220]. Note that Wang et al. [220] incorporate a bouncing mechanism and numerical approximations for the integration time to hit the failure barrier. Moreover, the results shown in Figure 6.4 improve on the coefficient of variation for the dimensions reported in [178], namely for  $d = 2, 5, 50, 100$ , with a reported lower bound of  $\delta = 0.3$ . As in Figure 6.3, both types of estimators agree. The most substantial difference is present in low dimensional settings. Whereas for high dimensions, the difference is negligible.

### 6.4.2 Multi-dimensional ring

This example is inspired by problems where there is interest in sampling from a specified region around a contour level of a given function. This particular problem can be encountered in History matching applications [4, 184]. The aim is to target a specific region of very low volume where one has statistical evidence



**Figure 6.4:** Results for the linear performance function in different dimensional settings  $d$ . The hard approximation uses the indicator functions for the computation of the coefficient of variation in (6.25). Whereas the smooth approximation refers to using the sigmoid (6.18) instead for such approximation.

of interest. The example that is presented here targets a ring with width of 0.5 units in different dimensional settings. The failure region is defined as

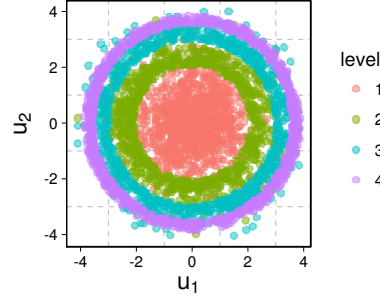
$$F = \left\{ \mathbf{u} \in \mathbb{R}^d : \left| \sum_{l=1}^d u_l^2 - r^2 \right| < 0.5 \right\}, \quad (6.28)$$

where  $r$  is the radius of the ring of interest. Assuming  $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, I)$ , a standard Gaussian, the sum of the squares is known to be distributed as a  $\chi_{(d)}^2$ . The ring is defined to be the 99.9% percentile of a chi-squared random variable. This has been chosen as it is the typical set of the corresponding multivariate Gaussian, [see 127, for a discussion of typical sets on standard Gaussian space]. Recall that SuS creates a sequence of decreasing values for the width of the ring,  $b = 0.5$ . Figure 6.5 shows that the SuS intermediate levels have two effects. First, the samples are pushed towards the ring of interest, the failure region. Secondly, it shrinks the width of the ring by selecting such a decreasing sequence for  $b$ .

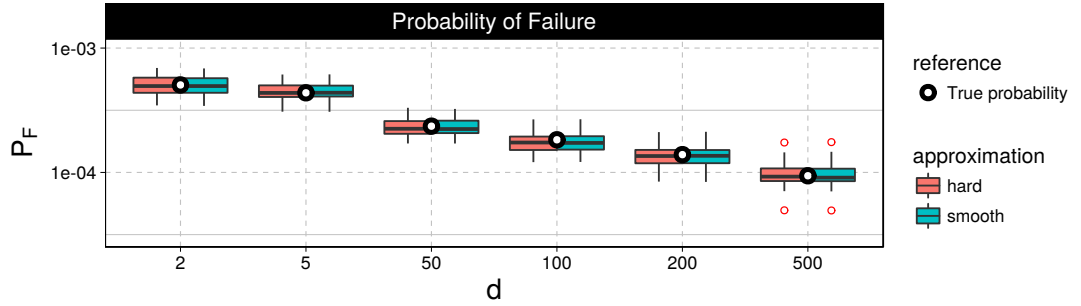
As before, Figure 6.6 shows the corresponding estimates as boxplots. This summarises 50 independent replications of the numerical experiment. The thick black dots show the true probability computed from the expression (6.28) for a chi-squared random variable. The results for the estimated failure probabilities summarised in Figure 6.6 show an accurate estimation across increasing dimensions.

The results shown for the coefficient of variation  $\delta$ , offer an additional insight. It can be noted that as the dimension increases the coefficient of variation seems to decrease. This suggests that the sampling method struggles to populate the





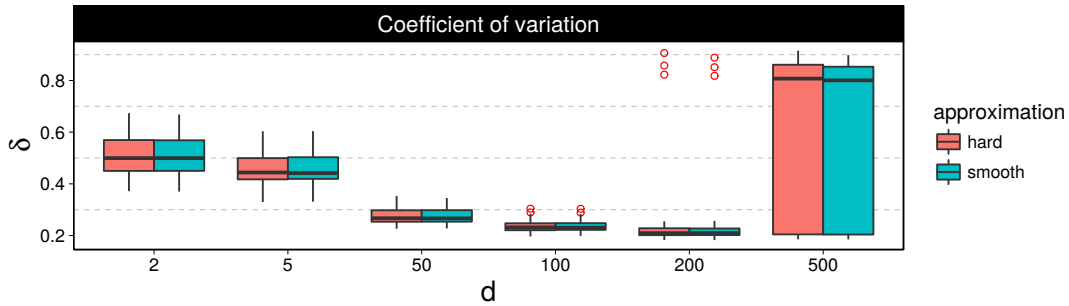
**Figure 6.5:** *Hamiltonian-based Subset simulation samples for the ring problem (6.28). Note that as the waves advance the ring becomes thinner. This is a consequence of the intermediate failure domain characterisation*



**Figure 6.6:** *Results for the ring performance function in different dimensional settings  $d$ . The hard approximation uses the indicator functions for the computation of the probability of failure in (6.7). Whereas the smooth approximation refers to using the sigmoid (6.18) instead for such approximation.*

target region as it becomes vacuous with increasing dimensionality. However, when dealing with the highest dimensional setting, namely  $d = 500$ , the coefficient of variation covers uniformly a larger range. Inspecting the results from the experiments, it is found that for  $d = 500$  and the outliers in  $d = 200$  the number of intermediate levels required were increased by one, in comparison to other dimensional settings. Recall, that  $\delta_M$  is a weighted sum of the  $k$ -lag autocorrelations  $\rho_M(k)$ . Higher values of these quantities indicate that the correlation between the *output* of the samples is increased. Recall that Hamiltonian-based sampling is an efficient tool to generate Markov Chains by suppressing random walk behaviour in  $\mathbf{u}$ . The motivation of using Hamiltonian sampling in Subset Simulation is to decrease the rejection rates at higher intermediate levels. Moreover, the little discrepancy between the individual  $\delta_M$  and the aggregate  $\delta$  show that for the intermediate levels  $\delta_j$  is kept small. The last level used in

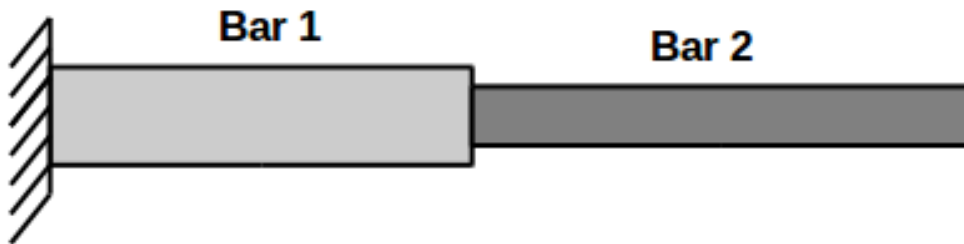
the experiments for  $d = 500$  was used to better populate the failure domain and achieved an overall good estimation of the probability of failure, see Figure 6.6.



**Figure 6.7:** Results for the ring performance function in different dimensional settings  $d$ . The hard approximation uses the indicator functions for the computation of the coefficient of variation in (6.25). Whereas the smooth approximation refers to using the sigmoid (6.18) instead for such approximation.

### 6.4.3 Inhomogeneous bar

This application is related to the static deflection of two connected bars (see Fig. 6.8) with random Young's modulus. The left bar has length  $L_1 = 0.12$  m and diameter  $D_1 = 7.5 \times 10^{-3}$  m. The right bar has length  $L_2 = 0.18$  m and section  $D_2 = 5 \times 10^{-3}$  m. A unit force is applied at the right end of the right bar.

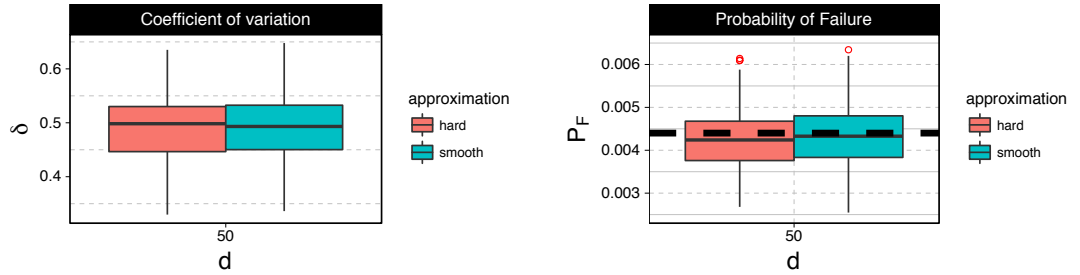


**Figure 6.8:** Graphical representation of the inhomogeneous bar.

We are interested in the right end deflection of the right bar when the Young's modulus is a random field. This random field is constructed from uniform Gaussian random field with squared exponential kernel with correlation length equal to 0.1, which is transformed to a log-normal random vector with mean value

$E = 7 \times 10^{10}$  Pa and coefficient of variation 0.05. The bars are discretized by mean of the finite element method with linear shape function. In each element, the Young's modulus is assumed to be constant. Each bar has 25 elements, resulting in a 50 dimensional problem.

The target level for the deflection was set to  $15.85 \times 10^{-3}$  m. For this experiment, there is no analytic expression for the gradient. Thus, it was computed with finite differences using a tolerance of  $10^{-8}$ . Additionally, it is important to remark some characteristics of the output of the simulator. Overall, the output shows little variation to changes in the input variables  $\mathbf{u}$ . The estimated standard deviation of the simulator output was calculated using 1,000 samples from  $\pi(\cdot)$ , a 50-dimensional multivariate Gaussian, and reported a value of  $\hat{\sigma} = 0.1148731$ . This shows a tight distribution for the simulator output. As a consequence, a higher temperature in the sigmoid approximation is needed. This was not the case for the previous experiments. Furthermore, recall the discussion about low variation in the simulator output from Figure 6.2b. Following this, the temperature was set to  $t = 200$  and the results are shown in Figure 6.9. As before, 50 independent runs of the Hamiltonian-based subset simulation were performed. The summary includes boxplots for the estimate of the probability of failure and the coefficient of variation, using both hard and smooth estimators. There is no indication of discrepancy among each pair, for both estimates.



**Figure 6.9:** Results for the dynamic model with random field. The underlying dynamic system was solved with a total of 50 elements. The boxplots summarise 50 independent replications of the Hamiltonian-based subset simulation. The thick-dashed line in the right panel shows the target probability of failure.

The true probability of failure is contained within the interquartile range, showing consistency in the estimates. It is important to note that deviations are contained within an order of magnitude and are, in average, accurate within the second significant digit. This shows evidence that no further computation

is needed by introducing additional intermediate failure domains. The reason is, that by setting  $p_0 = 0.1$ , each intermediate level improves the estimate within an order of magnitude, as can be seen from equation (6.7).

## 6.5 Discussion

Subset simulation is an efficient Markov chain Monte Carlo (MCMC) technique to be used in Reliability analysis problems where the target probability of failure is sufficiently small. In these cases, a typical rejection based sampling method would require a great amount of computation to estimate a rare event probability. Subset simulation alleviates the computational demand needed, by generating a sequence of intermediate regions that allow to estimate the probability as a product of more manageable domains. This has proven to be effective by decreasing the amount of simulator runs. Subset simulation also shows good performance with increasing dimensions, as it usually incorporates component-wise update to the target variables. However, as the intermediate failure domains needed become rarer, the geometry and complexity of the target region induces high rejection rates. The reason behind this is that the domain of interest is only added as rejection rule but is not actively participating on the proposal of samples.

In this chapter Hamiltonian Monte Carlo is proposed to be used to generate samples for the Markov chains needed in Subset Simulation. The failure domain is incorporated in the proposals by means of a sigmoid function. The latter, takes the role of a smooth barrier for the generation of samples during the simulation. The smoothness of the barrier alleviates the non-differentiability issues of incorporating the *failure* step in the generation of samples. The additional computational cost induced by the computation of the Hamiltonian trajectory improves as shown by measuring the coefficient of variation for the linear response experiment. Additionally, it is a natural candidate when dealing with highly constrained target regions, *e.g.* sampling from contour level sets.

Moreover, the smooth-barrier formulation allows an easy implementation of the algorithm in general purpose probabilistic programming software such as Stan or PyMC3. The computation of gradients in this framework is automatically done by automatic differentiation and support user-defined functions. This improves the probabilistic modelling effort and allows one to explore different approaches

for a particular problem. To finalize, to determine an effective strategy for the temperature parameter is still an open question and is the target of future research.

## 6.6 Conclusion

Hamiltonian-based subset simulation offers increased efficiency in the exploration and generation of samples for reliability analysis. In particular, it has proven to be successful in generating samples from highly constrained spaces as in the ringed failure domain experiment. As mentioned before, this direction was taken as the non-implausible space formulation is analogous to the failure domain characterisation in reliability analysis. An advantage of subset simulation with respect to the simulated annealing approach, and in particular to transitional MCMC, is that the number of chains is kept fixed throughout the algorithm. This offers a constant workload in parallel implementations. Moreover, the use of first order information allows one to improve not only the scaling of the proposal distribution, but also the directions in which to generate new samples. It should be noted that further computational reduction can be achieved if a surrogate model is used to approximate the performance function as discussed in Chapters 3 and 4. This was not pursued in this dissertation, but has been studied independently by [20, 119].

Subset simulation has been a successful sampling algorithm and it has inspired interesting directions of research. The next chapter studies a subset simulation-based approach for general Bayesian inference problems [206] as it exploits a formulation akin to rejection-based sampling. Chapter 7 develops a variant of the Bayesian Updating with Structural reliability method (BUS) that avoids the prior specification of a critical parameter for the algorithm.



# CHAPTER 7

---

## Reliability-based Calibration<sup>1</sup>

---

The previous chapter has studied the reliability analysis problem in engineering computations. It also provided an overview of the subset simulation algorithm to generate samples from negligible failure domains with higher computational efficiency. The study of reliability analysis has been motivated as an analogy can be drawn from the history matching formulation (Chapter 5) as discussed in Chapter 6. The success of subset simulation in high-dimensional reliability problems has motivated the development of techniques for more general settings. In particular, reliability-based sampling techniques have been developed by Straub and Papaioannou [206] to solve Bayesian inference (Chapter 2) applications in engineering computations such as the calibration problem discussed in Chapter 5. This chapter, presents a new automatic algorithm to be used in multimodal inferential tasks in Bayesian inference with parallelising capabilities.

### 7.1 Introduction

Making inference about the parameters of a mathematical model based on observed measurements of the real system is one of the most important problems in modern science and engineering. As discussed in Chapter 1, the Bayesian approach provides

---

<sup>1</sup>The results and ideas presented in this chapter have been published in DiazDelaO et al. [63]

a fundamental means to do this in the context of probability logic [148, 187, 121], where the parameters are viewed as uncertain variables and the inference results are cast in terms of their probability distribution after incorporating information from observed data, as discussed in Chapter 2. In engineering dynamics, for example, vibration data from a structure is collected from sensors and used for identifying the modal properties (*e.g.* natural frequencies, damping ratios, mode shapes) and structural model properties (*e.g.* stiffness, mass) [120, 75]. This has been formulated in a Bayesian context [17, 15], which resolved a number of philosophically challenging issues of the inverse problem, such as the treatment of multiple sets of parameters giving the same model fit to the data, an issue known as *identifiability*.

Let  $\mathbf{u} \in \mathbb{R}^n$  be a set of parameters of a model  $\mathcal{M}$ , based on which a probabilistic prediction of the data  $\mathcal{D}$  can be formulated through the likelihood function  $P(\mathcal{D}|\mathbf{u}, \mathcal{M})$ . As discussed in Chapter 2, the probability distribution of  $\mathbf{u}$  depends on the available information. Based only on knowledge in the context of  $\mathcal{M}$ , the distribution is described by the prior distribution  $P(\mathbf{u}|\mathcal{M})$ . When data about the system is available, it can be used to update the distribution. Using Bayes' Theorem, the posterior distribution that incorporates the data information in the context of  $\mathcal{M}$  is given by

$$P(\mathbf{u}|\mathcal{D}, \mathcal{M}) = P(\mathcal{D}|\mathcal{M})^{-1} P(\mathcal{D}|\mathbf{u}, \mathcal{M}) P(\mathbf{u}|\mathcal{M}), \quad (7.1)$$

where

$$P(\mathcal{D}|\mathcal{M}) = \int P(\mathcal{D}|\mathbf{u}, \mathcal{M}) P(\mathbf{u}|\mathcal{M}) d\mathbf{u}, \quad (7.2)$$

is a normalizing constant. Future predictions of a response quantity of interest, say  $r(\mathbf{u})$ , can be updated by incorporating data information, through the posterior expectation [176]:

$$E[r(\mathbf{u}|\mathcal{D}, \mathcal{M})] = \int r(\mathbf{u}) P(\mathbf{u}|\mathcal{D}, \mathcal{M}) d\mathbf{u}. \quad (7.3)$$

As far as the posterior distribution of  $\mathbf{u}$  for a given model  $\mathcal{M}$  is concerned, the constant in Eq. (7.2) is immaterial because it does not change the distribution. However, It is the primary quantity of study in Bayesian model class selection problems where competing models are compared based on the value of



$P(\mathcal{M})P(\mathcal{D}|\mathcal{M})$  [39, 41, 18]. In that context,  $P(\mathcal{D}|\mathcal{M})$  is often called the *evidence* (the higher the better).

Capturing efficiently essential information about the posterior distribution, *i.e.* posterior statistics, and calculating the posterior expectation is a non-trivial problem, primarily resulting from the complexity of the likelihood function. In many applications, the likelihood function is only implicitly known, *i.e.* its value can be calculated point-wise but its dependence on the model parameters is mathematically intractable. This renders analytical solutions infeasible and conventional numerical techniques inapplicable. In this case, Markov Chain Monte Carlo (MCMC) [154, 111, 188, 79] is found to provide a powerful computational tool. MCMC allows the samples of an arbitrarily given distribution to be efficiently generated as the samples of a specially designed Markov chain. In MCMC, candidate samples are generated by a *proposal distribution* (chosen by the analyst) and they are adaptively accepted based on ratios of the target distribution value at the candidate and the current sample.

While MCMC in principle provides a powerful solution for Bayesian computation, difficulties are encountered in applications, motivating different variants of the algorithm. For example, in problems with a large amount of data, the posterior distribution takes on significant values only in a small region of the parameter space, whose size generally shrinks in an inverse square root law with the data size. Depending on sufficiency or relevance of the data for the model parameters, the regions of significant probability content can be around a set of isolated points (globally or locally identifiable) or a lower dimensional manifold (unidentifiable) with non-trivial geometry [128, 129]. To the least extent this causes efficiency problems, making the choice of the proposal distribution difficult and leading to high rejection rate of candidates and hence poor efficiency. When the issue is not managed, significant bias can result in the statistical estimation based on the samples. Strategies similar to simulated annealing have been proposed to convert the original difficult updating problem effectively into a sequence of more manageable problems with less data, thereby allowing the samples to adapt gradually [16, 42, 45]. Another issue is *dimension sustainability*, *i.e.* whether the algorithm remains applicable when the number of variables (*i.e.* dimension) of the problem increases. This imposes restrictions on the design of MCMC algorithms so that quantities such as the ratio of likelihood functions involved in the simulation process do not *degenerate* as the dimension of the problem increases.

Application robustness and dimension sustainability are well-recognized in the engineering reliability method literature [10, 196, 127]. In this area, the general objective is to determine the failure probability that a scalar response of interest exceeds a specified threshold value, or equivalently to determine its complementary cumulative distribution function (CCDF) near the upper tail (*i.e.* large thresholds). As discussed in Chapter 6, Subset Simulation (SuS) [9, 12] has been developed as an advanced Monte Carlo strategy that is efficient for small failure probabilities (rare events) but still retain a reasonable robustness similar to the Direct Monte Carlo method. In SuS, samples conditional on a sequence of intermediate failure events are generated by MCMC and they gradually populate towards the target failure region. These *conditional samples* provide information for estimating the whole CCDF of the response quantity of interest. SuS typically does not make use of any problem-specific information, treating the input-output relationship between the response and the uncertain parameters as a *black box*. Based on an independent-component MCMC strategy, it is applicable for an arbitrary (potentially infinite) number of uncertain variables in the problem.

By establishing an analogy with the reliability problem that SuS is originally designed to solve, it is possible to adapt SuS to provide an efficient solution for another class of problems. For example, by considering an *augmented reliability problem* where deterministic design parameters are artificially considered as uncertain, SuS has been applied to investigate the sensitivity of the failure probability with respect to the design parameters and their optimal choice without repeated simulation runs [8, 46, 203, 211]. Another example can be found in constrained optimization problems, where an analogy was established between rare failure events in reliability problems and extreme events in optimization problems, allowing SuS to be applied to solving complex problems with nonlinear objective functions and potentially a large number of inequality constraints and optimization variables [141, 183].

In view of the application robustness and dimension sustainability, it would be attractive to adapt SuS for Bayesian computations. This is not trivial since the problem contexts are different. One major difference is that in the reliability problem the uncertain parameters follow standard classes of distributions (*e.g.* Gaussian, exponential) specified by the analyst; while in the Bayesian updating problem the uncertain parameters follow the posterior distribution, which generally does not belong to any standard distribution because the likelihood function is

problem-dependent.

Recent developments have shown promise for adapting SuS to Bayesian updating problems. In the context of Approximate Bayesian Computation (ABC), [44] built an analogy with the reliability problem so that the posterior samples in the Bayesian updating problem can be obtained as the conditional samples in SuS at the highest simulation level determined by a tolerance parameter that gradually diminishes. The latter controls the approximation of the likelihood function through a proximity model (a feature of ABC) between the measured and simulated data for a given value of model parameter.

Along another line of thought, [206] recently provided a formulation called BUS (Bayesian Updating using Structural reliability methods) that opens up the possibility of Bayesian updating using SuS. It combined an earlier idea [205] with the standard rejection principle to establish an analogy between a Bayesian updating problem and a reliability problem, or more correctly a *probabilistic failure analysis* problem [10, 7, 12]. Through the analogy, the samples following the posterior distribution in the Bayesian updating problem can be obtained as the conditional samples in the reliability problem. Unlike ABC, the formulation is exact as it respects fully the original likelihood function; and in this sense it is more fundamental. One outstanding problem, however, is the choice of the *likelihood multiplier*, or *multiplier* in short, in the context of rejection sampling methods. To guarantee the theoretical correctness of the analogy, it must be less than the reciprocal of the maximum value of the likelihood function, which is generally unknown especially before the problem is solved. Some suggestions have been given in [206] based on inspection of the likelihood function. An adaptive choice was suggested based empirically on the generated samples [30]. It is more robust to applications as it does not require prior input from the analyst. It offers no guarantee on correctness, however, due to the incomplete nature of finite sampling information which seems inevitable. The problem with the choice of the multiplier remains open.

The developments of this chapter are motivated by the choice of the multiplier and, more fundamentally, its mathematical and philosophical role in the BUS formulation. A rigorous mathematical study is carried out to provide fundamental understanding of the multiplier, which leads to a revised BUS formulation allowing SuS to be implemented independent of the choice of the multiplier and convergence of results to be checked formally. Essentially, by defining the failure event in the

BUS formulation, it is shown that the SuS-variant can in fact be implemented *without the multiplier* and the samples beyond a certain simulation level all have the same target posterior distribution.

This chapter is organized as follows. An overview of the original BUS formulation is first given. The mathematical role of the multiplier and its bias effect arising from inappropriate choice are then investigated. A revised formulation is then proposed and associated theoretical issues are investigated, followed by a discussion on the application of SuS under the revised formulation. Examples are presented to explain the theory and illustrate its applications.

## 7.2 BUS formulation

In this section a brief review of the BUS formulation is given [206, 207]. It builds an analogy between the Bayesian updating problem and a reliability problem, thereby allowing SuS to be applied to the former. For mathematical clarity and to simplify notation, in the Bayesian updating problem, let  $\pi(\mathbf{u})$  denote the prior PDF  $P(\mathbf{u})$ ,  $\mathcal{L}(\mathbf{u})$  the likelihood function  $P(\mathbf{u}|\mathcal{D}, \mathcal{M})$ ,  $P_{\mathcal{D}}$  the normalizing constant  $P(\mathcal{D}|\mathcal{M})$ , and  $\pi_{\mathcal{D}}(\mathbf{u})$  the posterior PDF. The same symbol  $\pi(\mathbf{u})$  is used for the prior PDF in the Bayesian updating problem and the parameter PDF in the reliability problem, as it has the same mathematical property (chosen from standard distributions by the analyst) and role (the distribution to start the SuS run) in both problems. In a Monte Carlo approach the primary target in Bayesian model updating is to generate samples according to the posterior PDF  $\pi_{\mathcal{D}}(\mathbf{u})$  (rewritten from (7.1)):

$$\pi_{\mathcal{D}}(\mathbf{u}) = P_{\mathcal{D}}^{-1} \pi(\mathbf{u}) \mathcal{L}(\mathbf{u}). \quad (7.4)$$

### 7.2.1 Rejection Principle

The BUS formulation is based on the conventional rejection principle. Let  $c$ , called the *likelihood multiplier* in this work, or simply *multiplier*, be a scalar constant such that for all  $\mathbf{u}$  the following inequality holds:

$$c \mathcal{L}(\mathbf{u}) \leq 1. \quad (7.5)$$

Also, assume that i.i.d. samples can be efficiently generated from the prior PDF

$\pi(\mathbf{u})$ . This is a reasonable assumption because the prior PDF is often chosen from a standard class of distributions (*e.g.* Gaussian, exponential). In the above context, a sample  $\mathbf{u}$  distributed as the posterior PDF  $\pi_{\mathcal{D}}(\mathbf{u}) \propto \pi(\mathbf{u})\mathcal{L}(\mathbf{u})$  in (7.4) can be generated from the following straightforward application of the rejection principle [61]:

Step 1. Generate  $U$  uniformly distributed on  $[0, 1]$  and  $\mathbf{u}$  distributed with the prior PDF  $\pi(\mathbf{u})$ .

Step 2. If  $U < c\mathcal{L}(\mathbf{u})$ , return  $\mathbf{u}$  as the sample. Otherwise go back to Step 1. It can be shown [206] that the sample  $\mathbf{u}$  returned from the above algorithm is distributed as  $\pi_{\mathcal{D}}(\mathbf{u})$ , that is by marginalising the auxiliary component  $u$  as

$$p_{\mathbf{u}}(\mathbf{u}) = \int_0^1 p_{\mathbf{u},u}(\mathbf{u}, u) du \propto \pi_{\mathcal{D}}(\mathbf{u}). \quad (7.6)$$

Although the above rejection algorithm is theoretically viable, the acceptance probability and hence efficiency is often very low in typical updating problems with a reasonable amount of data. This is because a sample drawn from the prior PDF  $\pi(\mathbf{u})$  often has a low likelihood value  $\mathcal{L}(\mathbf{u})$  when the data is informative about the uncertain parameters, leading to significant change from the prior to the posterior PDF.

### 7.2.2 Equivalent reliability problem

Recognizing the high rejection rate when the rejection principle is directly applied, BUS transforms the problem into a reliability problem. The premise is that this will allow the existing algorithms developed in the reliability method literature to be applied to Bayesian updating problems, especially those are that capable of generating samples from the frequent (safe) region to the rare (failure) region, such as SuS. The reliability problem analogy of the Bayesian updating problem is constructed as follows. Consider a reliability problem with uncertain parameters  $(\mathbf{u}, u)$  having the joint PDF  $\pi(\mathbf{u}) \mathbf{1}(0 \leq u \leq 1)$ , where the *failure event* is defined as

$$F = \{U < c\mathcal{L}(\mathbf{u})\}. \quad (7.7)$$

Suppose that by some means (*e.g.* SuS) a *failure sample* can be obtained, and is distributed as  $\pi(\mathbf{u}) \mathbf{1}(0 \leq u \leq 1)$  and conditional on the failure event  $F$ . The

PDF of the failure sample, denoted by  $(\mathbf{u}', U')$ , is given by

$$p_{\mathbf{u}', U'}(\mathbf{u}, u) = P_F^{-1} \pi(\mathbf{u}) \mathbf{1}(0 \leq u \leq 1) \mathbf{1}(u < c\mathcal{L}(\mathbf{u})), \quad (7.8)$$

where

$$P_F = \int \int \pi(\mathbf{u}) \mathbf{1}(0 \leq u \leq 1) \mathbf{1}(u < c\mathcal{L}(\mathbf{u})) du d\mathbf{u}, \quad (7.9)$$

is the *failure probability* of the reliability problem.

In the above formulation, the driving response variable can be defined as

$$Y = c\mathcal{L}(\mathbf{u}) - U, \quad (7.10)$$

so that the failure event corresponds to

$$F = \{Y > 0\}. \quad (7.11)$$

Populations of failure samples conditional on the intermediate failure events  $F_i = \{Y > b_i\}$  for adaptively increasing  $b_i$  ( $i = 1, 2, \dots$ ) are then generated until they pass the target failure event  $F = \{Y > 0\}$ , from which the samples conditional on  $F$  are collected as the posterior samples.

Note that in the original formulation the driving response variable was in fact defined  $Y = U - c\mathcal{L}(\mathbf{u})$ . The presentation in (7.10) is adopted so that it is consistent with the conventional SuS literature, where the intermediate threshold levels increase rather than decrease as the simulation level ascends.

## 7.3 Likelihood multiplier

One issue of concern in the BUS formulation is the choice of the multiplier  $c$  satisfying the inequality in (7.5), which is not always trivial. Some suggestions were given, by inspecting the mathematical structure of the likelihood function [206]; or by adaptively using empirical information from the generated samples [30]. The latter is more robust as it does not require preliminary analysis, but, as stated by the authors, in order to guarantee that it satisfies the inequality, more theoretical analysis is needed. In this section a rigorous investigation is performed to study the role of the multiplier and its effect on the results if it is not properly

chosen. The investigation leads to a reformulation of BUS, to be proposed in the next section.

In the context of BUS, the multiplier needs to be chosen before starting a SuS run as it affects the definition of the driving variable  $Y$  in (7.10). Clearly, the multiplier affects the distribution of the driving variable as well as the generated samples. Recall that only those samples conditional on  $Y = c\mathcal{L}(\mathbf{u}) - U > 0$  are collected as the posterior samples. The larger the value of  $c$  the more efficient the SuS run, because this will increase  $Y$  and the failure probability  $P(Y > 0)$ , thereby reducing the number of simulation levels required to reach the target failure event.

From the inequality in (7.5), the choice of the multiplier is governed by the region in the parameter space of  $\mathbf{u}$  where the value of  $\mathcal{L}(\mathbf{u})$  is large. The largest admissible value of  $c$  is given by

$$c_{\max} = [\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]^{-1}. \quad (7.12)$$

This result is well-known in the rejection sampling literature [61]. Clearly, this value is not known before computation. While using a value smaller than  $c_{\max}$  will be less efficient but still give the correct distribution in the samples, using a value larger than  $c_{\max}$  will lead to bias in the distribution of the samples. In some problems it is possible to investigate the mathematical structure of  $\mathcal{L}(\mathbf{u})$  and derive inequalities to propose a choice of  $c$  that guarantees  $c\mathcal{L}(\mathbf{u}) \leq 1$ . In such cases, it is computationally beneficial to use that value. However, in general it is difficult by numerical means to have a choice of  $c$  that guarantees the inequality.

When an inadmissible (too large) value of the multiplier is used, the resulting distribution of the failure samples will be truncated, leading to bias in the posterior statistical estimates based on them. To see this, note that the inequality (7.5) was used in establishing the third equality in (7.6). Suppose this inequality is violated, say, within some region  $B$ :

$$B = \{\mathbf{u} \in \mathbb{R}^n : c\mathcal{L}(\mathbf{u}) > 1\}. \quad (7.13)$$

Then for any  $\mathbf{u} \in B$ ,  $\mathbf{1}(u < c\mathcal{L}(\mathbf{u})) = 1$  for  $u \in (0, 1)$  and so (7.6) implies

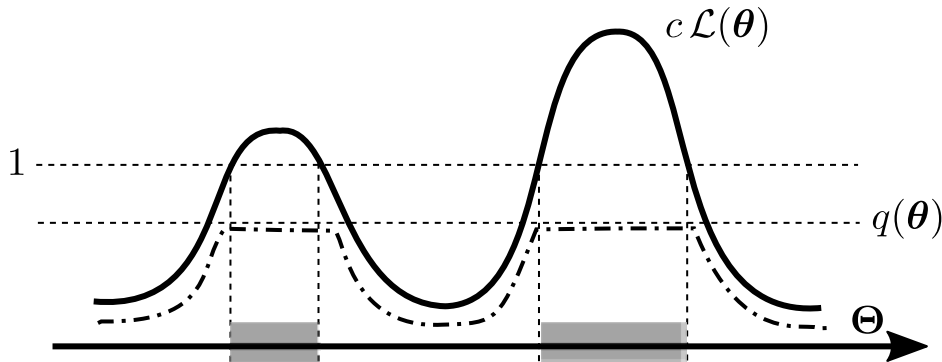
$$p(\mathbf{u}) = P_F^{-1} \pi(\mathbf{u}) \int_0^1 \mathbf{1}(u < c\mathcal{L}(\mathbf{u})) du = P_F^{-1} \pi(\mathbf{u}). \quad (7.14)$$

For those  $\mathbf{u}$  not in  $B$ , the inequality is satisfied and the PDF value  $p(\mathbf{u})$  remains to be the correct posterior PDF  $\pi_{\mathcal{D}}(\mathbf{u})$  as in (7.6):

$$p(\mathbf{u}) = P_F^{-1} \pi(\mathbf{u}) c\mathcal{L}(\mathbf{u}) \propto \pi_{\mathcal{D}}(\mathbf{u}). \quad (7.15)$$

Thus, an inadmissible (too large) value of  $c$  introduces bias in the problem by truncating the posterior PDF to be the prior PDF in the region of  $\mathbf{u}$  where the inequality is violated. Intuitively, in the context of rejection principle, if the multiplier is not small enough, the samples drawn from the prior PDF are accepted (incorrectly) *too often*, rendering their distribution closer to the prior PDF than they should be.

The truncation effect is illustrated in Figure 7.1, where the shaded interval denotes the region  $B$ . The prior PDF  $\pi(\mathbf{u})$  is taken to be constant and so  $\pi_{\mathcal{D}}(\mathbf{u}) \propto c\mathcal{L}(\mathbf{u})$ . Instead of the target posterior PDF, the resulting distribution of the sample takes the shape of the center line. Within the region  $B$  it is truncated to the shape of  $\pi(\mathbf{u})$ .



**Figure 7.1:** Truncation of distribution in rejection algorithm. Center line - resulting distribution (short of the constant  $P_F^{-1}$ ); shaded interval - truncation region  $B$  where  $c\mathcal{L}(\mathbf{u}) > 1$ . Figure used from [63], in the context of this dissertation,  $\theta = \mathbf{u}$  and  $q(\cdot) = \pi(\cdot)$

As long as the multiplier satisfies the inequality in (7.5), it is completely arbitrary and it does not affect the distribution of the resulting samples, which is equal to the correct posterior PDF. This observation is trivial but has important implications. In the original BUS context, for example, it implies that the samples generated in different simulation runs with different admissible values of the multiplier can be simply averaged for estimating posterior statistics, because they all have the same



correct posterior distribution. This fact shall also be used later when developing the proposed algorithm in this work.

## 7.4 Alternative BUS formulation

Having clarified the role of the multiplier, a modification of the original BUS formulation is presented. This isolates the effect of the multiplier in a fundamental manner. It also leads to a formulation where SuS can be performed without having to choose the multiplier before the simulation run; and where the effect of the multiplier appears clearly in the accuracy of the posterior distribution. The modification is based on the simple observation that the failure event in (7.7) can be rewritten as

$$F = \left\{ \ln \left[ \frac{\mathcal{L}(\mathbf{u})}{U} \right] > -\ln c \right\}. \quad (7.16)$$

This means that the driving variable in SuS can be defined as

$$Y = \ln \left[ \frac{\mathcal{L}(\mathbf{u})}{U} \right], \quad (7.17)$$

and the target failure event can now be written as

$$F = \{Y > b\}, \quad (7.18)$$

where

$$b = -\ln c. \quad (7.19)$$

The base of the logarithm is arbitrary but the natural logarithm is chosen here to facilitate the analysis.

Despite the apparently slight change in definition of the driving variable, the setup above changes the philosophy behind the multiplier and the way SuS is implemented to produce the posterior samples. The driving variable no longer depends on the multiplier and so the choice of the latter is no longer needed to start the SuS run. The multiplier only affects the target threshold level  $b$  beyond which the samples can be collected as posterior samples. As remarked at the end of the last section, as long as the multiplier is sufficiently small to satisfy the

inequality in (7.5), the distribution of the samples conditional on the failure event  $F = \{U < c\mathcal{L}(\mathbf{u})\}$  is invariably equal to the posterior distribution. This implies that in the proposed formulation the distribution of the samples conditional on  $\{Y > b\}$  will settle (remain unchanged) for sufficiently large  $b$ . In the original BUS formulation where the driving variable is defined as  $Y = c\mathcal{L}(\mathbf{u}) - U$  in (7.19) for a particular value of  $c$  (assumed to be admissible), only the samples conditional on the failure event  $F = \{Y > 0\}$ , *i.e.* for a threshold value of exactly zero, have the posterior distribution.

Substituting  $b = -\ln c$  from (7.19) into (7.5) and rearranging, the inequality constraint in terms of  $b$  is given by, for all  $\mathbf{u}$ ,

$$b > \ln \mathcal{L}(\mathbf{u}). \quad (7.20)$$

From (7.12), the maximum admissible value of  $c$  is  $c_{\max} = [\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]^{-1}$ . Correspondingly the minimum value of  $b$  beyond which the distribution of samples will settle at the posterior PDF is

$$b_{\min} = -\ln c_{\max} = \ln [\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]. \quad (7.21)$$

Similar to  $c_{\max}$ , the value of  $b_{\min}$  is generally unknown but this does not affect the SuS run. Under the proposed formulation, one can simply perform SuS with increasing levels until one determines that the threshold level of the highest level has passed  $b_{\min}$ . Despite not knowing  $b_{\min}$ , this turns out to be a more well-defined task as it is shown later that the CCDF of  $Y$ , *i.e.*  $P(Y > b)$  versus  $b$ , has characteristic behaviour for  $b > b_{\min}$ .

The logarithm in the above formulation is introduced for analytical and computational reasons, so that the driving variable is a well-defined random variable. In particular

$$Y = \ln \left[ \frac{\mathcal{L}(\mathbf{u})}{U} \right] = \ln \mathcal{L}(\mathbf{u}) + \ln(U^{-1}). \quad (7.22)$$

For  $U$  uniformly distributed on  $[0, 1]$ ,  $\ln(U^{-1})$  is exponentially distributed with mean 1. For a well-posed likelihood function  $\mathcal{L}(\mathbf{u})$  one can expect that  $\ln \mathcal{L}(\mathbf{u})$  is a well-defined random variable when  $\mathbf{u}$  is distributed as  $\pi(\cdot)$ , and so is the driving variable  $Y$ . In particular, if the first two moments of  $\ln \mathcal{L}(\mathbf{u})$  are bounded, then the same is also true for the first two moments of  $Y$  because

$$\begin{aligned} E[Y] &= E[\ln \mathcal{L}(\mathbf{u}) + \ln U^{-1}] \\ &= E[\ln \mathcal{L}(\mathbf{u})] + 1, \end{aligned} \quad (7.23)$$

$$\begin{aligned} E[Y^2] &= E\{[\ln \mathcal{L}(\mathbf{u}) + \ln U^{-1}]^2\} \\ &= E\{[\ln \mathcal{L}(\mathbf{u})]^2\} + 2E[\ln \mathcal{L}(\mathbf{u})]E[\ln U^{-1}] + E\{[\ln U^{-1}]^2\} \\ &= E\{[\ln \mathcal{L}(\mathbf{u})]^2\} + 2E[\ln \mathcal{L}(\mathbf{u})] + 2, \end{aligned} \quad (7.24)$$

since  $E[\ln U^{-1}] = 1$  and  $E\{[\ln U^{-1}]^2\} = 2$  (properties of the exponential variable  $\ln U^{-1}$ ).

It is argued that, while respecting the originality of BUS, the proposed formulation resolves the issue with the multiplier, as the requirement of choosing it a priori in the original formulation has been eliminated. The theoretical foundation of the proposed formulation is encapsulated in the following theorem.

**Theorem 6.** *Let  $\mathbf{u} \in \mathbb{R}^n$  be a random vector distributed as  $\pi(\mathbf{u})$  and  $U$  be a random variable uniformly distributed on  $[0, 1]$ ; with  $\mathbf{u}$  and  $U$  independent. Let  $\mathcal{L}(\mathbf{u})$  be a non-negative scalar function of  $\mathbf{u}$ . Define  $Y = \ln[\mathcal{L}(\mathbf{u})/U]$  and  $b = -\ln c$ , for  $c \in \mathbb{R}$ . Then, for any  $b > \ln[\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]$ :*

1. *The distribution of  $\mathbf{u}$  conditional on  $\{Y > b\}$  is  $\pi_{\mathcal{D}}(\mathbf{u}) = P_{\mathcal{D}}^{-1} \pi(\mathbf{u}) \mathcal{L}(\mathbf{u})$  where  $P_{\mathcal{D}} = \int \pi(z) \mathcal{L}(z) dz$  is a normalizing constant;*
2.  *$P_{\mathcal{D}} = e^b P(Y > b)$ .*

*Proof.* In order to prove the first part of the above theorem, first note that events  $\{Y > b\}$  and  $\{c\mathcal{L}(\mathbf{u}) > U\}$  are equivalent. Integrating out the uniform random variable from the PDF of the failure sample given by equation (7.8) gives:

$$\begin{aligned} p_{\mathbf{u}'}(\mathbf{u}) &= \int_0^1 p_{\mathbf{u}', U'}(\mathbf{u}, u) du \\ &= p_F^{-1} \pi(\mathbf{u}) \int_0^1 \mathbf{1}(0 \leq u \leq 1) \mathbf{1}(u < c\mathcal{L}(\mathbf{u})) du \\ &= p_F^{-1} \pi(\mathbf{u}) c\mathcal{L}(\mathbf{u}) \\ &\propto \pi_{\mathcal{D}}(\mathbf{u}). \end{aligned} \quad (7.25)$$

The result will be valid for any  $c < [\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]^{-1}$ , or equivalently for any  $b > \ln[\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]$ .

For the second part of the theorem, since  $Y = \ln[\mathcal{L}(\mathbf{u})/U]$  and  $(\mathbf{u}, U)$  has a joint PDF  $\pi(\mathbf{u})\mathbf{1}(0 < u < 1)$ ,  $P(Y > b)$  is given by

$$\begin{aligned} P(Y > b) &= \int \int \pi(\mathbf{u}) \mathbf{1}(0 < u < 1) I\left(\ln\left[\frac{\mathcal{L}(\mathbf{u})}{u}\right] > b\right) du d\mathbf{u} \\ &= \int \pi(\mathbf{u}) \int_0^1 \mathbf{1}(u < e^{-b}\mathcal{L}(\mathbf{u})) du d\mathbf{u} \\ &= e^{-b} \int \pi(\mathbf{u}) \mathcal{L}(\mathbf{u}) d\mathbf{u}, \end{aligned} \quad (7.26)$$

since  $\int_0^1 \mathbf{1}(u < e^{-b}\mathcal{L}(\mathbf{u})) du = e^{-b}\mathcal{L}(\mathbf{u})$  when  $e^{-b}\mathcal{L}(\mathbf{u}) < 1$  for all  $\mathbf{u}$  ( $b$  is admissible). Observe, from the definition of the posterior (7.1), that  $P_{\mathcal{D}}$  is simply the last integral in (7.25). Thus,

$$P_{\mathcal{D}} = e^b P(Y > b) \quad b > b_{\min}. \quad (7.27)$$

That is, when  $b > b_{\min}$ ,  $P_{\mathcal{D}}$  can be obtained as a product of  $e^b$  and the failure probability  $P(Y > b)$  it corresponds to. ■

## 7.5 Bayesian model class selection

In addition to providing the posterior distribution and estimating the updated expectation in (7.3), the posterior samples can be used for estimating the normalizing constant  $P_{\mathcal{D}}$  in (7.2). This is the primary target of computation in Bayesian model class selection problems, where competing models are rated. In this section it is shown how this can be done using the conditional samples generated by SuS in the context of the proposed formulation.

Let  $b$  be an admissible threshold level, *i.e.*  $b > b_{\min}$ , so that the samples conditional on  $\{Y > b\}$  have the correct posterior distribution  $\pi_{\mathcal{D}}(\mathbf{u})$ . Consider the failure probability  $P(Y > b)$ , which can be estimated using the samples in SuS.

Note that equation (7.27) can be rewritten as

$$P(Y > b) = e^{-b} P_{\mathcal{D}} \quad b > b_{\min}. \quad (7.28)$$

Since  $P_{\mathcal{D}}$  is constant for a given problem, this suggests that for sufficiently large  $b$ ,

$P(Y > b)$  will decay exponentially with  $b$ . Interpreting  $P(Y > b)$  as the CCDF of  $Y$ , this exponential decay gives a picture similar to a typical CCDF encountered in reliability analysis. This is another (though secondary) merit of introducing the logarithm in the definition of the driving variable  $Y$  in (7.17).

## 7.6 Characteristic trends

As shown in the last section, when  $b > b_{\min}$  the failure probability  $P(Y > b)$  is theoretically related to the evidence  $P_{\mathcal{D}}$  through (7.27). In the actual implementation,  $b_{\min}$  is not known and so it is necessary to determine whether  $b > b_{\min}$  so that the samples conditional on  $\{Y > b\}$  can be confidently collected as posterior samples. We argue that the variation of  $P(Y > b)$  with  $b$  takes on different characteristics on two different regimes of  $b$ . This can be used to tell whether the threshold value of a particular simulation level has already passed  $b_{\min}$  in a SuS run, thereby suggesting a stopping criterion.

First, note that  $P(Y > b)$  is a non-increasing function of  $b$ . When  $b$  is at the left tail of the CCDF,  $P(Y > b) \approx 1$  and it typically decreases with  $b$ , equal to  $P_{\mathcal{D}}$  at  $b > b_{\min}$ . When  $b > b_{\min}$ , it can be seen from (7.28) that  $P(Y > b) = P_{\mathcal{D}}e^{-b}$  and so it decays exponentially with  $b$ . We can thus expect that, as  $b$  increases from the left tail and passes  $b_{\min}$ , the CCDF of  $Y$  typically changes from a decreasing function to a fast (exponentially) decaying function. Correspondingly, the function  $\ln P(Y > b)$  changes from a slowly decreasing function to a straight line with a slope of -1.

On the other hand, consider the following function:

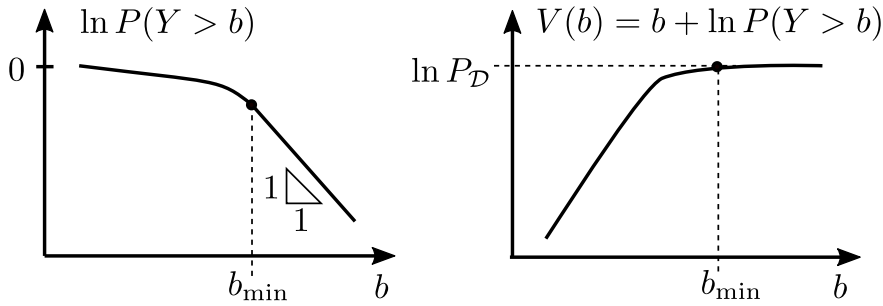
$$V(b) = b + \ln P(Y > b). \quad (7.29)$$

This function can be used for computing the log-evidence  $\ln P_{\mathcal{D}}$  as it can be readily seen that

$$V(b) = \ln P_{\mathcal{D}} \quad b > b_{\min}. \quad (7.30)$$

When  $b$  is at the left tail of the CCDF,  $\ln P(Y > b) \approx 0$  and so  $V(b) \approx b$  increases linearly with  $b$ . The above means that as  $b$  increases from the left tail of the CCDF of  $Y$  the function  $V(b)$  increases linearly, going through a transition until

it settles (remains unchanged) at  $\ln P_{\mathcal{D}}$  after  $b > b_{\min}$ . The characteristic behavior of  $\ln P(Y > b)$  and  $V(b)$  are depicted in Figure 7.2.



**Figure 7.2:** Characteristic trends of  $\ln P(Y > b)$  and  $V(b)$ .

Strictly speaking, the above arguments only apply to the theoretical quantities. In a SuS run the quantities  $\ln P(Y > b)$  and  $V(b)$  as a function of  $b$  can only be estimated on a sample basis. The resulting estimated counterparts will exhibit random deviation from the theoretical trends due to statistical estimation error, whose extent depends on the number of samples used in the simulation run (the larger the number of samples, the smaller the error). Nevertheless, the above arguments and Figure 7.2 provide the basis for determining the simulation level to stop and to collect the posterior samples, that is, once the transition in the slope of  $\ln P(Y > b)$  and  $V(b)$  is complete. On this basis, an automatic stopping condition is enforced, once the algorithm detects that the transition has occurred.

## 7.7 Automatic Stopping Strategy

In the proposed context, the posterior samples can be obtained from the conditional samples in a straightforward manner from a SuS run. No modification of SuS is necessary. The discussion below outlines how this can be done, focusing only on issues directly related to the Bayesian updating problem.

The primary target of the Bayesian updating problem is to generate posterior samples of  $\mathbf{u}$  distributed as the posterior PDF  $\pi_{\mathcal{D}}(\mathbf{u}) \propto \pi(\mathbf{u})\mathcal{L}(\mathbf{u})$ , where  $\pi(\mathbf{u})$  is the prior distribution assumed to be chosen from a standard class of distributions (e.g., Gaussian, exponential); and  $\mathcal{L}(\mathbf{u})$  is the likelihood function for a given set of data. As discussed in Chapter 6, a SuS run produces the estimate of the CCDF of the driving variable  $Y$ , *i.e.*  $P(Y > b)$  versus  $b$ . The posterior samples

for Bayesian model updating can be obtained as the conditional samples in a SuS run for the reliability problem with driving variable  $Y = \ln[\mathcal{L}(\mathbf{u})/U]$ , where  $\mathbf{u}$  is distributed as  $\pi(\mathbf{u})$  and  $U$  is uniformly distributed on  $[0,1]$ ; with  $\mathbf{u}$  and  $U$  independent. The conditional samples are collected from the level whose threshold level is determined to be greater than  $b_{\min}$ .

### 7.7.1 Stopping criterion

From the discussion in Section 7.6 and the definition of SuS, it is clear that the intermediate failure levels will continue to increase as the algorithm progresses. For a given level  $k$  where  $b_k$  is an admissible value for the failure event, the samples generated will eventually be distributed as desired. The following theorem establishes theoretical guarantees that such failure level can be achieved in a finite number of iterations, given some regularity assumptions. Moreover, it provides a stopping criterion to terminate the algorithm and prevent the generation of unnecessary SuS levels.

**Theorem 7.** *Let the Bayesian inference problem be defined by an upper-bounded likelihood function  $\mathcal{L}(\mathbf{u})$ , a prior density  $\pi(\mathbf{u})$  and associated posterior  $p(\mathbf{u}|\mathcal{D})$ . The marginal distribution of  $\mathbf{u}$  conditional on the intermediate failure levels, denoted by  $p(\mathbf{u}|F_k)$ , converges to the posterior. Moreover, there exist constants  $e^{-b_k}$  and a monotone decreasing sequence  $a_k$ , such that*

$$\lim_{k \rightarrow \infty} a_k = 0. \quad (7.31)$$

where  $a_k$  is the prior probability of the set  $B_k = \{\mathbf{u} : e^{-b_k} \mathcal{L}(\mathbf{u}) > 1\}$ .

*Proof.* In Theorem 6, it was proved that as long as the  $j$ -th failure level satisfies  $b_j > b_{\min}$ , any sample generated will be distributed according to the target posterior distribution. The level  $b_j$  is said to be a terminal level since any value of  $b_{j+1}$  is, by definition,  $b_{j+1} > b_j$ . Hence, the samples will be distributed as desired for any terminal level.

To prove the theorem, let us characterise a non-terminal level  $k$  such that  $b_k < b_{\min}$ . For the optimal threshold level  $b_{\min}$ , the inequality

$$u < e^{-b_{\min}} \mathcal{L}(\mathbf{u}) < 1, \quad (7.32)$$

is guaranteed for any value of a failure sample  $(\mathbf{u}, u)$  being distributed jointly as equation (7.8). In contrast, a non-terminal level satisfies  $e^{-b_{\min}} \mathcal{L}(\mathbf{u}) < e^{-b_k} \mathcal{L}(\mathbf{u})$  and it is not possible to determine an analogous right-hand side of inequality (7.32). Let the inadmissible set be defined as  $B_k = \{\mathbf{u} : e^{-b_k} \mathcal{L}(\mathbf{u}) > 1\}$ . It follows that the marginal distribution of the target variable is given by

$$p(\mathbf{u}|F_k) \propto \begin{cases} \pi(\mathbf{u}) & \text{if } \mathbf{u} \in B_k \\ e^{-b_k} \pi(\mathbf{u}) \mathcal{L}(\mathbf{u}) & \text{if } \mathbf{u} \in B_k^c. \end{cases} \quad (7.33)$$

Note that for all samples in the inadmissible set  $B_k$ , the marginal is proportional to the prior distribution, whilst for the samples in the admissible set  $B_k^c$  the target density is proportional to the posterior distribution. Marginalising in order to compute the normalising constant results in

$$\begin{aligned} P_{F_k} &= \int_{\Theta} [\pi(\mathbf{u}) I(\mathbf{u} \in B_k) + e^{-b_k} \pi(\mathbf{u}) \mathcal{L}(\mathbf{u}) I(\mathbf{u} \in B_k^c)] d\mathbf{u} \\ &= \int_{B_k} \pi(\mathbf{u}) d\mathbf{u} + e^{-b_k} \int_{B_k^c} \pi(\mathbf{u}) \mathcal{L}(\mathbf{u}) d\mathbf{u} \\ &= P_{\mathbf{u}}(B_k) + e^{-b_k} P_{\mathcal{D}} P_{\mathbf{u}|\mathcal{D}}(B_k^c), \end{aligned} \quad (7.34)$$

where  $P_{\mathbf{u}}(B_k)$  denotes the probability of event  $B_k$  under the prior distribution and  $P_{\mathbf{u}|\mathcal{D}}(B_k^c)$  denotes the probability of event  $B_k^c$  under the posterior distribution. Note that equation (7.34) is consistent with the case where  $b_k$  is a terminal level. If that is the case, the pair  $(\mathbf{u}, u)$  satisfies  $u < e^{-b_k} \mathcal{L}(\mathbf{u})$  by the definition of the driving variable  $Y$  and thus  $B_k = \emptyset$ . Let us rewrite the inadmissible set as

$$B_k = \{\mathbf{u} : \mathcal{L}(\mathbf{u}) > e^{b_k}\}. \quad (7.35)$$

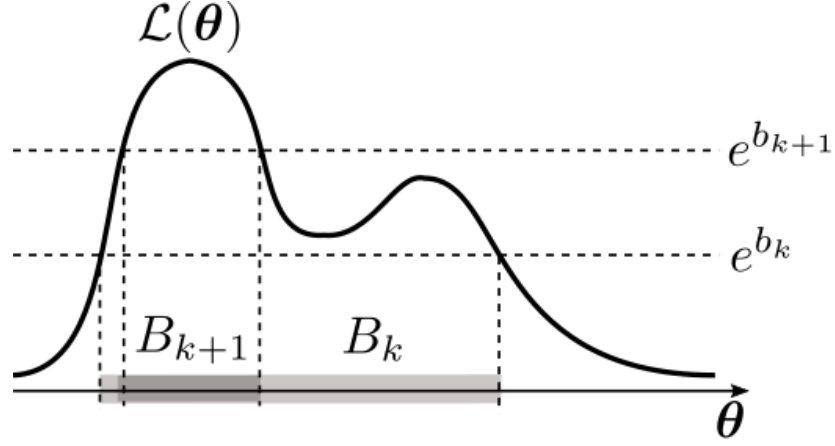
Given an increasing sequence of failure levels, it can be seen that the sequence of inadmissible sets is monotone decreasing, namely

$$B_k \supset B_{k+1} \supset \dots \supset \emptyset. \quad (7.36)$$

This fact is depicted in Figure 7.3.

Additionally, since the prior distribution is a probability measure, it satisfies the monotonicity property, namely  $P(B_{k+1}) \leq P(B_k)$  for all  $k$ . Let us define the sequence  $a_k$  as the prior probability of the inadmissible sets, *i.e.*  $a_k = P_{\mathbf{u}}(B_k)$ .





**Figure 7.3:** Increasing failure levels and likelihood.

As a consequence of the monotonicity property, it follows that  $a_k$  is a monotone decreasing sequence of values converging to zero from above, denoted by

$$a_k \searrow 0. \quad (7.37)$$

Moreover, since the sets  $B_k$  are monotone decreasing, then the sequence of complements is increasing, that is

$$B_k^c \subset B_{k+1}^c \subset \dots \subset \Theta. \quad (7.38)$$

Let  $m_k$  denote the posterior probability of the set  $B_k^c$ . Analogous to  $a_k$ , the sequence  $m_k$  is monotone increasing converging to 1 from below. This is denoted by

$$m_k \nearrow 1. \quad (7.39)$$

Expressions (7.37) and (7.39) allow to establish that for a sufficiently large value of  $k$

$$p_{F_k} = e^{-b_k} P_{\mathcal{D}}, \quad (7.40)$$

is satisfied and the result is established. ■

The preceding theorem allows us to propose a stopping criterion for the BUS

algorithm with driving variable  $Y = \log[\mathcal{L}(\mathbf{u})/u]$  using SuS . The value of  $a_k$  can be made arbitrarily small by means of the failure level  $b_k$ , which is learnt automatically during the algorithm. The computation of  $a_k$  is challenging, since it involves a multiple integral. Note that the prior probability can be written as

$$a_k = P_{\mathbf{u}}(B_k) = P_{\mathbf{u}}(\mathcal{L}(\mathbf{u}) > e^{b_k}) \quad (7.41)$$

which is in itself a reliability problem, where the likelihood  $\mathcal{L}(\mathbf{u})$  takes the role of a performance function and  $e^{b_k}$  is a reliability threshold. Since the prior distributions are chosen from a standard catalogue of density functions and the probability is assumed to be small, such integral can also be computed by means of SuS. In this setting, computing equation (7.41) can be regarded as performing an *inner level* SuS. The sampling of the expanded variables  $(\mathbf{u}, u)$  from the failure levels in equation (7.8), is regarded as *outer level* SuS.

### 7.7.2 Posterior statistical estimation

The posterior samples  $\{\mathbf{u}_k^{(m)} : k = 1, \dots, N\}$  obtained from simulation level  $m$  for which  $b_m > b_{\min}$  can be used for estimating posterior statistics in Bayesian updating problem and the evidence for Bayesian model class section. For the former, the posterior expectation in (7.3) is estimated by simple averaging:

$$E[r(\mathbf{u})|\mathcal{D}, \mathcal{M}] \approx \frac{1}{N} \sum_{k=1}^N r(\mathbf{u}_k^{(m)}). \quad (7.42)$$

On the other hand, based on (7.27), the evidence can be estimated by

$$P(\mathcal{D}|\mathcal{M}) = P_{\mathcal{D}} \approx \hat{P}_{\mathcal{D}} = e^{b_m} p_0^m. \quad (7.43)$$

Taking logarithm, the log-evidence is estimated by

$$\ln P(\mathcal{D}|\mathcal{M}) = \ln P_{\mathcal{D}} \approx \ln \hat{P}_{\mathcal{D}} = b_m + m \ln p_0. \quad (7.44)$$

### 7.7.3 Statistical error assessment

Some comments are in order regarding the statistical error of the results, in terms of the quality of the posterior samples and the statistical variability of the log-evidence estimator. Provided that the threshold value of the simulation

level is greater than  $b_{\min}$ , its conditional samples are always distributed as the target posterior PDF  $\pi_{\mathcal{D}}(\mathbf{u})$ . As MCMC samples they are correlated, however. When used for statistical estimation they will give less information compared to if they were independent. Typically their correlation tends to increase with the simulation level. In view of this, it is not necessary to perform more simulation levels than necessary. The stopping criterion based on the inner-outer procedure discussed above guards against this scenario.

For the evidence estimate in (7.43), it should be noted that its statistical variability arises from  $b_m$ . By taking small random perturbation of the estimation formula, it can be reasoned that  $\text{c.o.v.}(\ln P) \approx \text{std}(\ln P) \approx \text{std}(b_m)$ , where  $\text{std}$  is an abbreviation for standard deviation. An estimation formula for the c.o.v. of  $b_m$  based on samples in a single SuS run is not available, however. Conventionally only the c.o.v. of the estimate  $\hat{P}_b$  (say) for  $P(Y > b)$  for fixed  $b$  is available, rather than the c.o.v. of the  $b$  quantile value  $b_m$  for fixed exceedance probability. It can be reasoned, however, that the c.o.v. of  $\hat{P}_{\mathcal{D}}$  (where  $b_m$  is random) can be approximated by the c.o.v. of  $e^b \hat{P}_b$  for fixed  $b$  (then taking  $b = b_m$  obtained in a simulation run). The latter is equal to the c.o.v. of  $\hat{P}_b$ , for which standard estimation formula is available [9, 12].

#### 7.7.4 Comparison with original BUS formulation

Table 1 provides a comparison between the original BUS and the proposed formulation. Implementing SuS under the proposed framework has several advantages over the original BUS, stemming mainly from the treatment of the multiplier in the former. First of all, there is no need to determine the appropriate value of the multiplier to start the simulation run. The definition of the driving variable is more intrinsic as it only depends on the likelihood function and not on the multiplier. In the BUS context, if the chosen value of the multiplier is not small enough, it will lead to bias in the distribution of the samples, unfortunately in the high likelihood region of the posterior distribution that is most important. If it is chosen too small it will result in lower efficiency, as it requires more simulation levels to reach the target event from which the samples can be taken as posterior samples. In both cases if it is found after a SuS run that the choice of the multiplier is not appropriate, one needs to perform an additional run with a (hopefully) better choice of the multiplier. These issues are all irrelevant in the proposed

context because the problem specification of the SuS run does not depend on the multiplier.

	BUS	Proposed
Driving variable	$Y = c\mathcal{L}(\mathbf{u}) - U$ for any $c < [\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]^{-1}$	$Y = \ln[\mathcal{L}(\mathbf{u})/U]$
Target failure event	$F = \{Y > 0\}$	$F = \{Y > b\}$ for any $b > \ln[\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]$
Evidence calculation	$P_{\mathcal{D}} = cP(Y > 0)$	$P_{\mathcal{D}} = e^b P(Y > b)$ for any $b > \ln[\max_{\mathbf{u}} \mathcal{L}(\mathbf{u})]$
Stopping criterion	When threshold value of simulation level is equal to zero.	After inner-outer SuS procedure automatically determines that the threshold $b_{\min}$ has been crossed by driving the sequence $a_k \searrow 0$ .

**Table 7.1:** Comparison of original BUS and proposed reformulation. Note that the original definition of the driving variable in BUS is  $Y = U - c\mathcal{L}(\mathbf{u})$ . For consistency with SuS literature, it has been reexpressed as shown here.

On the other hand, in the BUS context the posterior samples must be obtained as those conditional on the target failure event  $\{Y > 0\}$  where  $Y = c\mathcal{L}(\mathbf{u}) - U$ . For example, samples conditional on  $Y > 0.1$  cannot be directly used. Since the threshold values  $b_1, b_2, \dots$  generated adaptively in different simulation levels of SuS are random, they generally do not coincide with 0, *i.e.* the target threshold value of interest. In this case, not all samples can be used directly as conditional samples. In the original BUS algorithm if the threshold level of the next level determined adaptively from the samples of the current level is greater than zero, it is set equal to zero so that the next (and final) level is exactly conditional on  $\{Y > 0\}$ . In the proposed context, the posterior samples can be directly collected from the samples generated in SuS. This is because any sample conditional on  $\{Y > b\}$  with  $b > b_{\min}$  ( $Y = \ln[\mathcal{L}(\mathbf{u})/U]$ ) can be taken as a posterior sample. The value of  $b_{\min}$  is unknown but whether  $b > b_{\min}$  can be determined from the inner-outer procedure discussed in Section 7.7.

## 7.8 Numerical Experiments

In this section, two examples are shown that illustrate the applicability of the proposed methodology. The first one is the locally identifiable case of a two-degree-of-freedom shear building model originally presented in [16]. The second example is the unidentifiable case of the same model.

### 7.8.1 Example 1. Two-DOF shear frame: locally identifiable case

Consider a two-storied building structure represented by a two-degree-of-freedom shear building model. The objective is to identify the interstory stiffnesses which allow the structural response to be subsequently updated. The first and second story masses are given by  $16.5 \times 10^3$  kg and  $16.1 \times 10^3$  kg respectively. Let  $\mathbf{u} = [\theta_1, \theta_2]$  be the stiffness parameters to be identified. The interstory stiffnesses are thus parameterized as  $k_1 = \theta_1 \bar{k}_1$  and  $k_2 = \theta_2 \bar{k}_2$ , where the nominal values for the stiffnesses are given by  $k_1 = k_2 = 29.7 \times 10^6$  N/m. The joint prior distribution  $\pi(\cdot)$  for  $\theta_1$  and  $\theta_2$  is assumed to be the product of two Lognormal distributions with most probable values 1.3 and 0.8 respectively and unit standard deviations. For further details on the assumptions behind the parameterization and the choice of nominal values, refer to [16]. Let  $\mathcal{D} = \{\tilde{f}_1, \tilde{f}_2\}$  be the modal data used for the model updating, where 3.13 Hz and 9.83 Hz are the identified natural frequencies. The posterior PDF is formulated following [214] as

$$\pi_{\mathcal{D}} \propto \exp[-J(\mathbf{u})/2\epsilon^2] \pi(\mathbf{u}), \quad (7.45)$$

where  $\epsilon$  is the standard deviation of the prediction error and  $J(\mathbf{u})$  is a modal measure-of-fit function given by

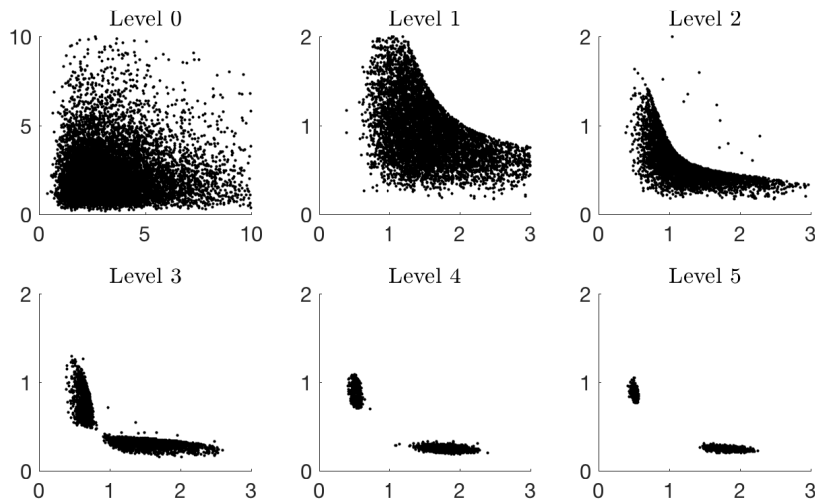
$$J(\mathbf{u}) = \sum_{j=1}^2 \lambda_j^2 [f_j^2(\mathbf{u})/\tilde{f}_j^2 - 1]. \quad (7.46)$$

Here,  $\lambda_1$  and  $\lambda_2$  are weights and  $f_1(\mathbf{u})$  and  $f_2(\mathbf{u})$  are the modal frequencies predicted by the corresponding finite element model.

For the implementation of SuS, a conventional choice of algorithm parameters in the reliability literature is adopted in this study. The level probability is chosen

to be  $p_0 = 0.1$  and the number of samples per level  $N$  is fixed at 10,000. In the standard Gaussian space, the one-dimensional proposal PDF is chosen to be uniform with a maximum step width of 1. A relatively large number of samples per level is been chosen in this study to illustrate the theoretical aspects of the proposed method. Strategies for efficiency improvement such as adaptive proposal PDF or likelihood function can be explored but are not further investigated here.

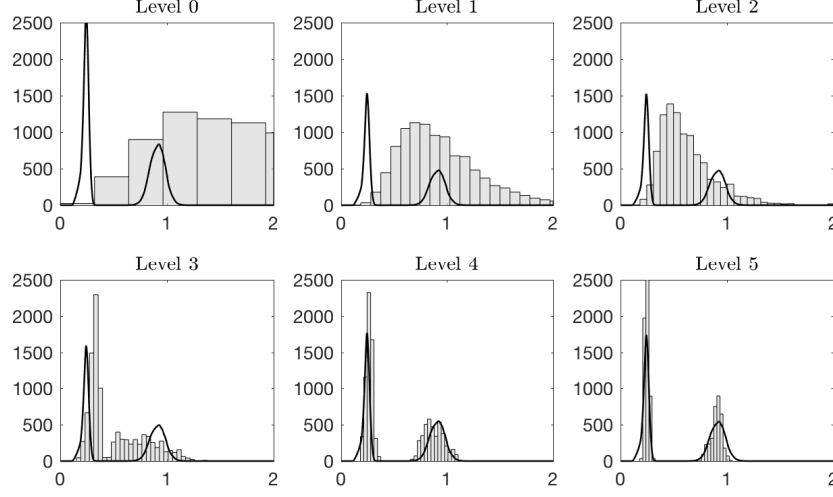
Figure 7.4 shows the Markov chain samples for  $\mathbf{u} = [\theta_1, \theta_2]$  at six consecutive simulation levels. The results are shown in the Lognormal space after the application of the relevant transformation. Level 0 corresponds to the unconditional case (*i.e.* Direct Monte Carlo), that is, the joint prior PDF. As the simulation level ascends, the distribution of the samples evolves from the prior distribution to the target posterior distribution, which is bimodal in the present example.



**Figure 7.4:** Markov chain samples in the Lognormal space for the stiffness parameters  $\theta = [\theta_1, \theta_2]$  from Level 0 (prior distribution) to Level 5.

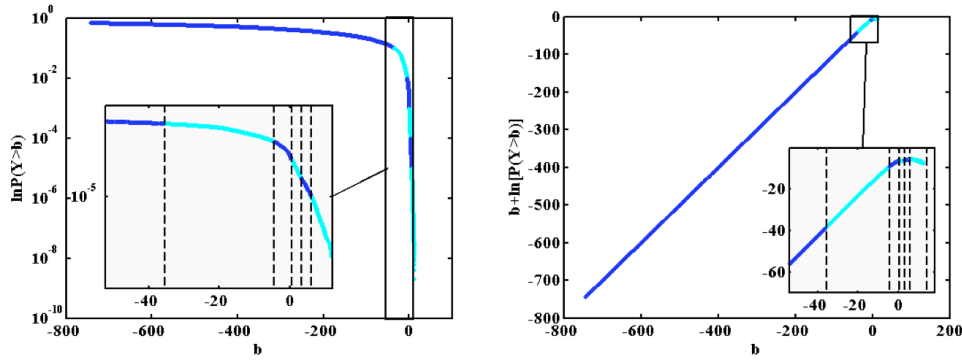
Figure 7.5 shows the marginal histograms for  $\theta_1$  and  $\theta_2$  corresponding to those samples in Figure 7.4. For comparison, the solid lines show the target marginal posterior distributions obtained by numerically integrating the expression for the posterior PDF, which is still feasible for this two-dimensional example. It is apparent that the distribution of the samples has settled either in Level 4 or Level 5. In reality, the exact target PDF is not available and so alternative means must be employed to determine whether the distribution of the samples has settled at the target one. Within the context of the current methodology, this is done

through the proposed automatic stopping strategy and confirmed by the plots of the log-failure probability and log-evidence versus the threshold level.



**Figure 7.5:** *Posterior marginal PDF for  $\theta_2$  at different simulation levels. The target marginal posteriors were obtained numerically and are shown for comparison.*

Figure 7.6 plots the estimate of the log-CCDF of  $Y$ , *i.e.*  $\ln P(Y > b)$  versus  $b$ . The general shape of the resulting simulated curve coincides with the characteristic trend predicted by the theory (see Figure 7.2), that is, there is a transition from a slowly decreasing function to a line with slope equal to -1. When zooming into the region where  $b > 0$ , the figure shows the boundaries of each level computed via SuS. Additionally, the log-evidence was computed following (7.29) and is shown in Figure 7.6. As with the log-CCDF, the theoretical prediction of the characteristic trend is also verified for this case, whereby the curve flattens when the transition is complete. Table 7.2 shows the evolution of the threshold (columns 2 and 3). The transition is complete after Level 4, where the probability of inadmissibility  $a_k$  converges to zero (as defined in Section 7.7). For a tolerance of  $a_k = 10^{-8}$ , the fourth column in Table 7.2 shows that the posterior samples should be collected from Level 5. This corresponds with the clearly bimodal distributions in figures 7.4 and 7.5. It is guaranteed that the samples in the subsequent SuS levels would all be distributed according to the target posterior PDF. However, for statistical estimation their quality deteriorates as the simulation level ascends because their correlation tends to increase. Thus, the algorithm stops in Level 5.



**Figure 7.6:** Log-CCDF computed through SuS (left plot) for the identifiable case. The curve slowly transitions into a straight line with negative unit slope. Correspondingly, the log-evidence (right plot) flattens as the threshold exceeds  $b_{\min}$ . The dotted lines show the thresholds for different simulation levels.

Level	$b_k$	$c_k$	$a_k$
0			
1	-4.291e+02	2.325e+186	5.3300e-01
2	-6.237e+01	1.221e+27	1.3800e-01
3	-9.331e+00	1.128e+04	2.8700e-02
4	2.203e+00	1.105e-01	4.0400e-03
5	5.780e+00	3.088e-03	0.0000e+00

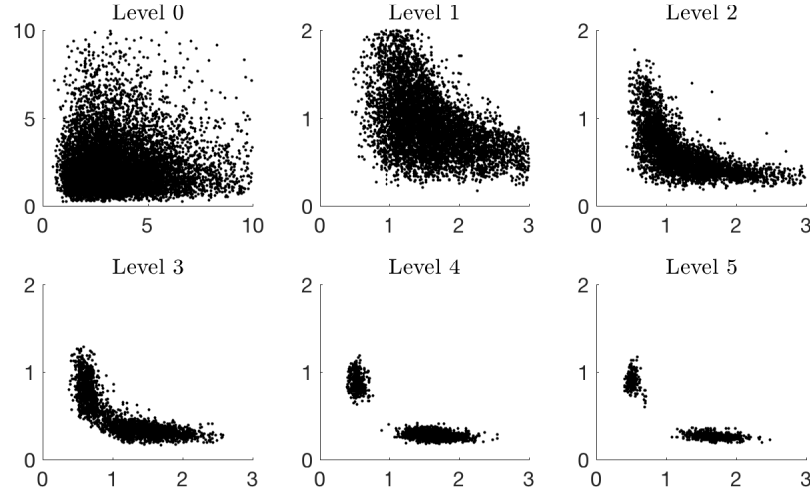
**Table 7.2:** Evolution of the threshold and the probability of inadmissibility.

### 7.8.2 Example 2. Two-DOF shear frame: unidentifiable case

The exercise was repeated for the case where the story masses are also unknown and need to be updated. The problem is characterized as unidentifiable, since there are an infinite number of combinations of parameter values that can explain the measured modal frequencies. In addition to the stiffnesses, the masses are parameterized as  $m_1 = \theta_3 \bar{m}_1$  and  $m_2 = \theta_4 \bar{m}_2$ , where the nominal values for the are given by  $m_1 = 16.5 \times 10^3$  kg and  $m_2 = 16.1 \times 10^3$  kg. Thus, for this case,  $\mathbf{u} = [\theta_1, \theta_2, \theta_3, \theta_4]$  where the marginal prior distributions for  $\theta_1$  and  $\theta_2$  are the same Lognormals as in the previous example. The prior marginal distributions for  $\theta_3$  and  $\theta_4$  are both assumed to be Lognormals with most probable values equal to 0.95 and standard deviation of 0.1. The joint prior PDF is therefore taken as



the product of the four Lognormals. Figure 7.7 shows the Markov chain samples for  $\mathbf{u}$  ( $\theta_1$  versus  $\theta_2$  for visualization purposes) at simulation levels 0 through 5. Again, the updated distribution results in a bimodal posterior PDF.

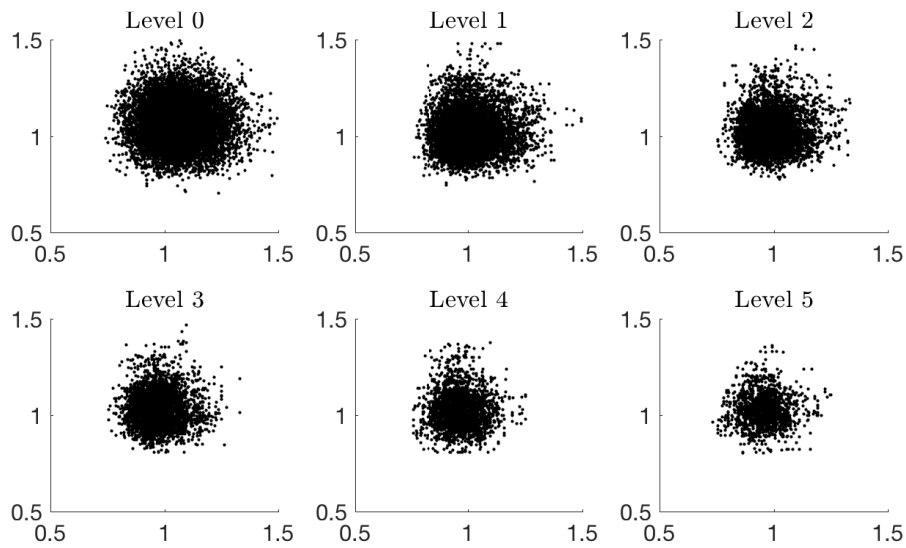


**Figure 7.7:** Markov chain samples in the Lognormal space for the stiffness parameters  $\theta_1$  and  $\theta_2$  of the unidentifiable case at simulation levels 0 (prior distribution) to level 5.

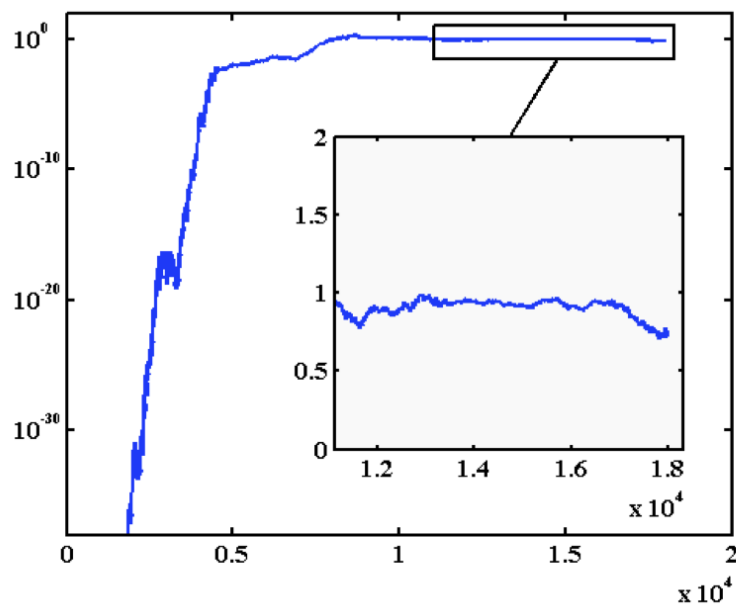
Analogously, Figure 7.8 shows the samples for  $\theta_3$  and  $\theta_4$  in the Lognormal space. There is no noticeable pattern in the distribution of the masses, consistent with the findings in [16]. The characteristics of this example are very similar to the ones displayed by the locally identifiable case. The automatic stopping condition is also reached when  $a_k \leq 10^{-8}$ , for which the posterior samples are also collected in Level 5. We omit the characteristic trend plots and corresponding table for brevity.

### 7.8.3 Example 3. Model Class Selection

Following the two preceding examples, the log-evidence corresponding to each model can be estimated according to equation (7.44). Figure 7.9 shows the ratio of the evidence for the identifiable case to the evidence of the locally unidentifiable case. Discounting the random deviation due to simulation error, the ratio of evidence seems to converge to 1, which suggests that, given the available data, there is no reason to prefer the unidentifiable model over the more parsimonious one.



**Figure 7.8:** Markov chain samples in the Lognormal space for the mass parameters  $\theta_3$  and  $\theta_4$  of the unidentifiable example at simulation levels 0 to level 5.



**Figure 7.9:** Ratio of evidence of the identifiable model to the evidence of the locally unidentifiable model. Since this ratio converges to 1, there is no preference of either model over each other, given the available data.

## 7.9 Discussion

A fundamental analysis of BUS, a recently proposed framework that establishes an analogy between the Bayesian updating problem and the engineering reliability problem, has been presented. This work was motivated by the question of choosing the correct likelihood multiplier and it has led to an improved formulation which resolves this question. By redefining the target failure event, the driving variable in the equivalent reliability problem has been expressed using the likelihood function alone, without the multiplier. This redefinition provides the key advantage over the original BUS, since the proposed implementation no longer requires a predetermined value for the multiplier in order to start the SuS runs. This immediately eliminates the need to perform additional runs in case an inadmissible or inefficient value for the multiplier is chosen. Moreover, it was shown that the samples generated at different levels of SuS can be used directly as posterior samples as long as their threshold is greater than the minimum admissible value and the probability of inadmissibility is zero. The proposed algorithm incorporates an inner-outer SuS procedure that provides an automatic stopping condition. The theoretical predictions of this study have been verified by applying the proposed strategy to illustrative examples.

## 7.10 Conclusion

This chapter presented a variant of the BUS algorithm which can be implemented without prior specification of the critical parameter. That is, the proposed algorithm can be implemented in an automatic fashion without the need to monitor the global trends or define *a priori* the likelihood multiplier. The formulation in this chapter has allowed the connection of reliability analysis with Bayesian inference (Chapter 2) in the context of engineering computations. Moreover, due to the analogy to the history matching formulation the algorithm developed in this chapter is akin to solve calibration problems as in Chapter 5.

The proposed algorithm is based on subset simulation which was previously introduced in Chapter 6. Future research directions include using the Hamiltonian-based subset simulation (studied in Chapter 6) to further increase the efficiency of the sampling method. This would allow one to overcome complex and small geometries of the support of the posterior, now seen as a non-improbable domain.

The reason is that in the more general context of Bayesian inference, the failure domain is equivalent to the support of a, possibly multimodal, posterior. Moreover, in BUS applications additional computational savings could be achieved if, for example, a Gaussian process is used to emulate the likelihood. That is the likelihood function of the Bayesian model could be treated as an expensive computer code as in Chapters 3 and 4. This could potentially be useful in Bayesian inference models where the posterior is computationally expensive to evaluate. The use of surrogate models in a Bayesian inference problem as developed in this chapter draws an additional analogy with the history matching problem. This is because this can also be understood as targeting non-implausible regions (support of the posterior) with a surrogate for the performance function (the likelihood). To follow this direction further lies out of the scope of this dissertation. However, it is conjectured that it would require careful study of the surrogate for the gradients needed in the numerical simulation of the Hamiltonian system defined in Chapter 6.

## CHAPTER 8

---

### Summary and Conclusions

---

The work carried out in this dissertation was focused on the development of full probabilistic characterisations to solve fundamental Uncertainty Quantification problems. In summary, three fundamental problems in engineering have been treated and new algorithms have been developed. Firstly, probabilistic methods have been developed to be used in surrogate modelling of computationally expensive computer codes. Secondly, the calibration problem of highly expensive computer codes has been addressed by a full probabilistic characterisation of history matching. Thirdly, the estimation of failure probabilities in reliability analysis has been addressed by incorporating an efficient sampling algorithm in highly constrained settings. Moreover, reliability-based sampling methods have been redefined for automatic implementations to be used in more general Bayesian inference problems. Individual chapters offer a summary of individual results and contributions. In this chapter the main findings and ideas are summarised, possible future research directions are provided, and a list of resulting publications is presented.

## 8.1 Summary of Completed Work

Chapter 1 opened the discussion with an introduction to Uncertainty Quantification (UQ) and the applications of interest for this dissertation. It provided a common ground for probabilistic reasoning for UQ tasks and introduced the problem of emulating computationally expensive computer codes, history matching and reliability analysis (RA). It presented a brief overview of surrogate models from two different perspectives. Namely, regression-based approaches and Sobol-based representations.

Chapter 2 provided an overview on Bayesian inference and Gaussian processes. The former allows for a framework to update beliefs of uncertain events in light of collected data. The latter provides a flexible probabilistic approach to be employed in surrogate modelling. Both topics served as the foundational aspects of the work developed in this dissertation. Bayesian inference provides a solid mathematical framework to update probabilities. In particular, this is used in the assimilation of observed data for complex modelling scenarios. Bayesian inference has also motivated the use of sampling algorithms to report inferences and results in simulation-based methods. In particular, Markov chain Monte Carlo has driven the widespread use of Bayesian methodology in science. On the other hand, Gaussian processes provide an elegant solution to the specification of probabilistic distributions in functional spaces. Thus, it allows to formulate prior knowledge in the functions of interest when the simulator output is assumed an uncertain quantity except for a collection of observed data.

Chapter 3 gave a detailed overview of Gaussian processes as surrogate models for computationally expensive computer codes. The presentation introduced Gaussian processes from a full Bayesian setting, which means that emphasis was placed on marginalising the hyperparameters of the model. This contrasts with the optimisation approach which is usually taken in the engineering literature. The reason for the widespread use of the maximum likelihood and maximum a posteriori estimates is mainly a consequence of computational convenience. The likelihood and posterior distribution are widely known to be multimodal, and it has been observed that predictive capabilities of a GP do not change drastically with posterior sampling. However, in applications with small datasets, sampling from the posterior distribution of the hyperparameters allows for a more robust quantification of uncertainty. This is due to the fact that multiple local

minima represent different types of surrogates. In particular, Chapters 3 and 4 introduced two new sampling algorithms for GP hyperparameter marginalisation. The foundations of these new sampling algorithms are simulated annealing and transitional MCMC. This allows to grow different Markov Chains, potentially in parallel. The proposed PAIMS algorithm introduces an adaptive component to improve the scaling of the proposals, as well as a delayed rejection mechanism to stimulate better mixing. The proposed TA<sup>2</sup>S<sup>2</sup> algorithm improves on the delayed rejection principle by introducing slice sampling as the transition operator. Under slice sampling, adaptive scaling and delayed rejection are seemingly integrated which improves upon sampling in the presence of multimodal distributions.

Computer codes that can run efficiently are not enough to tackle real-world engineering challenges. The trust that can be deposited into these numerical models relies on how well they are calibrated. Chapter 5 provides a probabilistic treatment of history matching. In particular, history matching allows to discard regions of parameter space where there is confidence the simulator is likely to replicate the data. This changes the perspective of looking for the best candidate within input space, in the sense that it provides the best-calibrated model, to seek and identify unacceptable configurations. This in turn, alleviates the highly expensive demands of the experimental process with an explicit characterisation that actively avoids placing too much trust in a single candidate. For this, a full Bayesian Gaussian process is used instead of a linear Bayes characterisation for the surrogate model used in history matching. This allows to incorporate the output of the surrogate as a random variable in the implausibility function. The algorithm developed and introduced in Chapter 4 is incorporated in its two modalities. First, a full Bayesian Gaussian process is proposed as an emulator for computationally expensive codes. The sampling is done with TA<sup>2</sup>S<sup>2</sup> targeting the posterior distribution. Second, the exploration and identification of the non-implausible region is done with TA<sup>2</sup>S<sup>2</sup> as in subset optimisation. The result is a collection of samples with the highest probabilistic implausibility measure. Furthermore, to refocus the emulator, three active learning criteria are proposed. These learning criteria explore the NROY space in a *one-look-ahead* manner.

The sampling problem in history matching has motivated the study of sampling algorithms used for reliability analysis. The reason is that the failure domain characterisation in reliability analysis is analogous to the non-implausible formulation in history matching. In reliability analysis, Subset Simulation has proven to

be an effective sampling method for rare events estimation. However, in heavily constrained failure domains, the sampling mechanism encounters high rejection rates in the later stages of the procedure. This is because of complex geometries and small volume of the failure region.

In Chapter 6, Hamiltonian Monte Carlo was proposed as an alternative transition proposal for subset simulation. The Hamiltonian-based Subset Simulation improved the exploration of the sampling space. Moreover, it allowed to incorporate the failure domain characterisation within the sampling algorithm. This means, that samples are generated and guided with first order information from the performance function. Thus, the performance function loses its discriminator-only nature, and becomes a more active ingredient of the simulation. The flexibility of the approach can be easily incorporated in existing probabilistic programming software. This in turn, allows improved experimentation with different probabilistic models.

The use of Subset Simulation in reliability analysis inspired new sampling algorithms for general Bayesian inference problems. For example, the BUS algorithm is a reformulation of rejection based sampling with a subset simulation component. Chapter 7 explores the BUS algorithm and changes the formulation to learn sequentially its critical parameter. A theoretical result provides an algorithmic check as a stopping condition.

## 8.2 Summary of Contributions

1. **Efficient sampling of GP hyperparameters.** As discussed in Chapter 3 the posterior probability function of the hyperparameters of a GP is known to be multimodal if not enough training data is collected. In Chapter 4 an automatic adaptive scheme was proposed and implemented, which can be implemented in parallel. This improves the efficiency of MCMC samplers as they are typically not parallel. Additionally, the efficiency of sample generation was solved using a slice sampled proposal. The flexibility of such methods allows the algorithms to be incorporated in other settings such as optimisation subtasks encountered in calibration of computer codes.
2. **Emulation-based history matching.** Calibration of expensive simulators to scarce data presents a challenge that can be addressed formally by



Bayesian updating. A full probabilistic characterisation of the GP as an emulator was exploited in Chapter 5. This provided robustness in quantifying uncertainty stemming from scarce experimental data. The flexibility of the sampling schemes presented in Chapters 3 and 4 allow to identify regions of parameter space where the simulator is likely to provide a match to experimental data, thus solving the calibration problem without over-tuning. In these regions the aim is to further improve the GP emulator. The selection of new simulation runs was formulated as an active learning task. Several variants of learning choices are presented in Chapter 5 within the context of history matching. This improved data acquisition and assimilation into the GP emulator.

3. **Reliability-based calibration.** The use of sampling schemes in engineering applications has increased dramatically over the years in different areas of engineering computations. In particular, reliability analysis is a subfield of engineering that aims to study the conditions where a system performs given specific standards. Through reliability analysis the analyst can determine the probability of a system to enter a failure state, an event of

non-zero but small probability. Subset simulation was employed to provide efficient generation of samples in topologically complex failure regions by means of an iterative procedure. Thus in Chapter 6, Hamiltonian Monte Carlo was proposed to smooth the reliability problem. This diminished the increasing rejection rates in subset simulation, and avoided setting the integration time by coupling it with a probabilistic programming framework. Inspired by Subset Simulation, reliability-based sampling methods aim to generate samples from a Bayesian inference problem by an auxiliary reformulation. Finally, in Chapter 7, a reformulation and theoretical stopping condition was developed in the context of Bayesian updating with reliability-based methods. In particular, the proposed algorithm incorporates rejection-based principles and is able to solve the calibration problem posed as a robust Bayesian inference procedure.

## 8.3 Outlook

There are several research directions that can be followed from the work in this dissertation. In the case of history matching the simulators might provide multiple outputs. The probabilistic formulation in Chapter 5 motivates an analogue study of multi-output probabilistic implausibility measures. Also, the learning criteria proposed can be seen as one-step-look ahead criteria. This opens the possibility of batch active learning methods to be used in history matching to refocus the emulator training.

The Hamiltonian-based subset simulation algorithm incorporates a parameter which resembles a temperature parameter in simulated annealing. It controls how close the smooth conditional density is to the original piecewise intermediate density. It is of great interest to provide an algorithmic proposal for the correct estimation of such parameters and avoid biased results in the estimation of failure probabilities. Furthermore, this development in Subset Simulation and the automatic BUS formulation provides a viable solution to multimodal sampling problems. Thus, it is of interest to extend to a Hamiltonian-based BUS algorithm that can be used in general Bayesian inference problems.

## 8.4 Published journal papers

1. **A. Garbuno-Inigo**, F. A. DiazDelaO, and K. M. Zuev. Transitional annealed adaptive slice sampling for gaussian process hyper-parameter estimation. *International Journal for Uncertainty Quantification*, 6(4):341359, 2016.
2. **A. Garbuno-Inigo**, F. A. DiazDelaO, and K. M. Zuev. *Gaussian process hyper-parameter estimation using Parallel Asymptotically Independent Markov Sampling*. *Computational Statistics & Data Analysis*, 103:367–383, 2016.
3. F. DiazDelaO, **A. Garbuno-Inigo**, S. Au, and I. Yoshida. Bayesian updating and model class selection with subset simulation. *Computer Methods in Applied Mechanics and Engineering*, 317:11021121, 2017.

## 8.5 Journal Papers Under Review

1. **A. Garbuno-Inigo**, F. A. DiazDelaO, and K. M. Zuev. Emulation-based history matching. *Under preparation*, 2018.
2. **A. Garbuno-Inigo**, F. A. DiazDelaO, A. Batou, and K. M. Zuev. Hamiltonian-based Subset Simulation. *Under preparation*, 2018.



---

## Bibliography

---

- [1] R. J. Adler and J. E. Taylor. *Random fields and geometry*. Springer Science & Business Media, 2009. [38](#), [41](#)
- [2] I. Andrianakis and P. G. Challenor. The effect of the nugget on Gaussian process emulators of computer models. *Computational Statistics and Data Analysis*, 56(12):4215–4228, 2012. [46](#), [50](#), [64](#), [66](#), [68](#), [82](#), [95](#)
- [3] I. Andrianakis, I. R. Vernon, N. McCreesh, T. J. McKinley, J. E. Oakley, R. N. Nsubuga, M. Goldstein, and R. G. White. Bayesian History Matching of Complex Infectious Disease Models Using Emulation: A Tutorial and a Case Study on HIV in Uganda. *PLoS Computational Biology*, 11(1): e1003968, 2015. [107](#), [111](#), [114](#)
- [4] I. Andrianakis, I. Vernon, N. McCreesh, T. McKinley, J. Oakley, R. Nsubuga, M. Goldstein, and R. White. History matching of a complex epidemiological model of human immunodeficiency virus transmission by using variance emulation. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 2016. [111](#), [153](#)
- [5] Y. Andrianakis and P. Challenor. Parameter estimation for gaussian process emulators. *Technical report, Managing Uncertainty in Complex Models*, 2011. [52](#)

- [6] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 72(3):269–342, 2010. [46](#), [91](#)
- [7] S. Au. Probabilistic failure analysis by importance sampling markov chain simulation. *Journal Of Engineering Mechanics*, 130(3):303–311, 2004. [165](#)
- [8] S. K. Au. Reliability-based design sensitivity by efficient simulation. *Computers & Structures*, 83(14):1048–1061, 2005. [164](#)
- [9] S.-K. Au and J. L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4):263–277, Oct. 2001. [19](#), [138](#), [140](#), [143](#), [151](#), [164](#), [181](#)
- [10] S.-K. Au and J. L. Beck. Important sampling in high dimensions. *Structural Safety*, 25(2):139–163, Apr. 2003. [164](#), [165](#)
- [11] S.-K. Au and E. Patelli. Rare event simulation in finite-infinite dimensional space. *Reliability Engineering & System Safety*, 148:67–77, 2016. [144](#), [147](#)
- [12] S.-K. Au and Y. Wang. *Engineering Risk Assessment with Subset Simulation*. Wiley, 2014. [18](#), [138](#), [140](#), [164](#), [165](#), [181](#)
- [13] L. S. Bastos and A. O’Hagan. Diagnostics for Gaussian process emulators. *Technometrics*, 51(4):425–438, 2009. [66](#), [68](#), [74](#), [75](#), [92](#)
- [14] J. Beck and K. M. Zuev. Asymptotically Independent Markov Sampling: a new MCMC scheme for Bayesian Inference. *International Journal for Uncertainty Quantification*, 3(5), 2013. [46](#), [55](#), [85](#), [91](#), [114](#)
- [15] J. L. Beck. Bayesian system identification based on probability logic. *Structural Control and Health Monitoring*, 17(7):825–847, 2010. [162](#)
- [16] J. L. Beck and S.-K. Au. Bayesian Updating of Structural Models and Reliability using Markov Chain Monte Carlo Simulation. *Journal of Engineering Mechanics*, 128(4):380–391, 2002. [163](#), [183](#), [187](#)
- [17] J. L. Beck and L. S. Katafygiotis. Updating models and their uncertainties. i: Bayesian statistical framework. *Journal of Engineering Mechanics*, 124(4):455–461, 1998. [162](#)

- 
- [18] J. L. Beck and K.-V. Yuen. Model selection using response measurements: Bayesian probabilistic approach. *Journal of Engineering Mechanics*, 130(2):192–203, 2004. [163](#)
- [19] J. Bect, D. Ginsbourger, L. Li, V. Picheny, and E. Vazquez. Sequential design of computer experiments for the estimation of a probability of failure. *Statistics and Computing*, 22(3):773–793, 2012. [109](#), [116](#)
- [20] J. Bect, L. Li, and E. Vazquez. Bayesian subset simulation. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):762–786, 2017. [4](#), [159](#)
- [21] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003. [11](#)
- [22] J. O. Berger and J. M. Bernardo. On the development of reference priors. *Bayesian Statistics*, 4(4), 1992. [51](#)
- [23] J. O. Berger, B. Liseo, and R. L. Wolpert. Integrated Likelihood Methods for Eliminating Nuisance Parameters. *Statistical Science*, 14(1):1–28, 1999. [54](#)
- [24] J. O. Berger, J. M. Bernardo, and D. Sun. The formal definition of reference priors. *Annals of Statistics*, 37(2):905–938, 2009. [26](#), [51](#)
- [25] J. O. Berger, J. M. Bernardo, D. Sun, et al. Overall objective priors. *Bayesian Analysis*, 10(1):189–221, 2015. [26](#)
- [26] A. Berlinet and C. Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011. [38](#), [41](#)
- [27] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. Wiley Series in Probability and Statistics, 2001. [24](#), [41](#)
- [28] D. P. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific, 2005. [118](#)
- [29] M. Betancourt. A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*, 2017. [145](#)

- [30] W. Betz, I. Papaioannou, and D. Straub. Adaptive variant of the BUS approach to Bayesian updating. (July):3021–3028, 2014. [165](#), [168](#)
- [31] J. R. Birge and N. G. Polson. Optimisation via Slice Sampling. *arXiv preprint*, pages 1–22, 2012. [82](#)
- [32] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006. [30](#)
- [33] G. Blatman and B. Sudret. Sparse polynomial chaos expansions and adaptive stochastic finite elements using a regression approach. *Comptes Rendus Mécanique*, 336(6):518–523, 2008. [14](#)
- [34] G. Blatman and B. Sudret. Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230(6):2345–2367, 2011. [14](#)
- [35] M. A. Bouhlef, N. Bartoli, A. Otsmane, and J. Morlier. Improving kriging surrogates of high-dimensional design models by partial least squares dimension reduction. *Structural and Multidisciplinary Optimization*, 53(5):935–952, 2016. [126](#)
- [36] J.-M. Bourinet, F. Deheeger, and M. Lemaire. Assessing small failure probabilities by combined subset simulation and support vector machines. *Structural Safety*, 33(6):343–353, 2011. [144](#)
- [37] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng. *Handbook of markov chain monte carlo*. CRC press, 2011. [29](#)
- [38] E. Capiez-Lernout and C. Soize. Robust design optimization in computational mechanics. *Journal of Applied Mechanics*, 75(2):021001, 2008. [130](#)
- [39] B. P. Carlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 473–484, 1995. [163](#)
- [40] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, A. Riddell, et al. Stan: A Probabilistic



- Programming Language. *Journal of Statistical Software*, 76(i01), 2017. [28](#), [139](#), [146](#), [147](#), [151](#)
- [41] M.-H. Chen, Q.-M. Shao, and J. G. Ibrahim. *Monte Carlo methods in Bayesian computation*. Springer Science & Business Media, 2012. [163](#)
- [42] S. H. Cheung and J. L. Beck. Calculation of Posterior Probabilities for Bayesian Model Class Assessment and Averaging from Posterior Samples Based on Dynamic System Data. *Computer-Aided Civil and Infrastructure Engineering*, 25(5):304–321, Feb. 2010. [163](#)
- [43] C. Chevalier and D. Ginsbourger. Fast computation of the multi-points expected improvement with applications in batch selection. In *Learning and Intelligent Optimization*, pages 59–69. Springer, 2013. [128](#)
- [44] M. Chiachio, J. L. Beck, J. Chiachio, and G. Rus. Approximate bayesian computation by subset simulation. *arXiv preprint arXiv:1404.6225*, 2014. [165](#)
- [45] J. Ching and Y.-C. Chen. Transitional Markov Chain Monte Carlo method for Bayesian model updating, model class selection and model averaging. *Journal of Engineering Mechanics*, 133(7):816–832, 2007. [45](#), [46](#), [62](#), [82](#), [91](#), [92](#), [163](#)
- [46] J. Ching and Y. Hsieh. Approximate reliability-based optimization using a three-step approach based on subset simulation. *Journal of Engineering Mechanics*, 133(4):481–493, 2007. [164](#)
- [47] D. CireşAn, U. Meier, J. Masci, and J. Schmidhuber. Multi-column deep neural network for traffic sign classification. *Neural networks*, 32:333–338, 2012. [11](#)
- [48] P. S. Craig, M. Goldstein, a. H. Seheult, and J. a. Smith. Bayes linear strategies for matching hydrocarbon reservoir history. *Bayesian Statistics 5*, pages 69–95, 1996. [4](#), [16](#), [17](#), [110](#), [116](#)
- [49] P. S. Craig, M. Goldstein, A. H. Seheult, and J. A. Smith. Pressure matching for hydrocarbon reservoirs: a case study in the use of Bayes linear strategies

- for large computer experiments. In *Case studies in Bayesian statistics*, pages 37–93. Springer, 1997. [4](#), [17](#), [43](#), [110](#), [116](#)
- [50] N. Cressie and C. K. Wikle. *Statistics for spatio-temporal data*. John Wiley & Sons, 2015. [4](#)
- [51] N. A. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, 1993. [4](#), [43](#), [108](#)
- [52] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416):953–963, 1991. [4](#)
- [53] V. De Oliveira. Objective Bayesian analysis of spatial data with measurement error. *Canadian Journal of Statistics*, 35(2):283–301, 2007. [66](#)
- [54] B. DeFinetti. *Theory of Probability: A Critical Introductory Treatment*. Wiley, Chichester, 1990. [16](#), [24](#)
- [55] P. Del Moral and A. Jasra. Sequential Monte Carlo for Bayesian Computation. *Bayesian Statistics*, 8:1–34, 2007. [91](#), [102](#)
- [56] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society B*, 68(3):411–436, 2006. [46](#), [86](#)
- [57] P. Del Moral, A. Doucet, and A. Jasra. On adaptive resampling strategies for sequential Monte Carlo methods. *Bernoulli*, 18(1):252–278, 2012. [46](#), [87](#)
- [58] V. Demyanov, L. Backhouse, and M. Christie. Geological feature selection in reservoir modelling and history matching with multiple kernel learning. *Computers & Geosciences*, 85:16–25, 2015. [7](#)
- [59] D. den Hertog, J. P. Kleijnen, and A. Siem. The correct Kriging variance estimated by bootstrapping. *Journal of the Operational Research Society*, 57(4):400–409, 2006. [94](#)
- [60] A. Der Kiureghian and P.-L. Liu. Structural reliability under incomplete probability information. *J. of Engr. Mech., ASCE*, 112(1):84–105, 1986. [138](#)

- 
- [61] L. Devroye. *Non-Uniform Random Variate Generation*. Springer, 1986. [167](#), [169](#)
- [62] P. Diaconis. Bayesian numerical analysis. *Statistical decision theory and related topics IV*, 1:163–175, 1988. [31](#)
- [63] F. DiazDelaO, A. Garbuno-Inigo, S. Au, and I. Yoshida. Bayesian updating and model class selection with subset simulation. *Computer Methods in Applied Mechanics and Engineering*, 317:1102–1121, 2017. [161](#), [170](#)
- [64] F. A. DiazDelaO and S. Adhikari. Gaussian process emulators for the stochastic finite element method. *International Journal for Numerical Methods in Engineering*, 87(6):521–540, 2011. [108](#)
- [65] I. Doltsinis and Z. Kang. Robust design of structures using optimization methods. *Computer Methods in Applied Mechanics and Engineering*, 193(23-26):2221–2237, June 2004. [130](#)
- [66] A. Doucet, N. De Freitas, and N. Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001. [29](#)
- [67] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer, 2001. [29](#), [46](#)
- [68] D. Draper. Assessment and propagation of model uncertainty. *Journal of the Royal Statistical Society, Series B*, 57(1):45–97, 1995. [44](#)
- [69] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987. [144](#)
- [70] N. Durrande, D. Ginsbourger, and O. Roustant. Additive kernels for gaussian process modeling. *arXiv preprint arXiv:1103.4023*, 2011. [36](#)
- [71] N. Durrande, J. Hensman, M. Rattray, and N. D. Lawrence. Gaussian process models for periodicity detection. *arXiv preprint arXiv:1303.7090*, 2013. [36](#)
- [72] D. K. Duvenaud, H. Nickisch, and C. E. Rasmussen. Additive gaussian processes. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira,

- and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 226–234. Curran Associates, Inc., 2011. [36](#)
- [73] B. Echard, N. Gayton, M. Lemaire, and N. Relun. A combined importance sampling and kriging reliability method for small failure probabilities with time-demanding numerical models. *Reliability Engineering & System Safety*, 111:232–240, 2013. [109](#), [117](#)
- [74] B. Eric, N. D. Freitas, and A. Ghosh. Active preference learning with discrete choice data. In *Advances in neural information processing systems*, pages 409–416, 2008. [109](#)
- [75] D. Ewins. *Modal testing*. Research Studies Press, Baldock, 2000. [162](#)
- [76] C. R. Farrar and K. Worden. *Structural Health Monitoring: A Machine Learning Perspective*. John Wiley & Sons, 2012. [7](#)
- [77] P. Fearnhead and B. M. Taylor. An adaptive sequential Monte Carlo sampler. *Bayesian Analysis*, 8(2):411–438, 2013. [60](#), [86](#)
- [78] W. Feller. *An introduction to probability theory and its applications*, volume 1. Wiley, New York, 1968. [24](#)
- [79] G. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications*. Springer, New York Berlin Heidelberg, 1996. [163](#)
- [80] A. I. J. Forrester, A. Sóbester, and A. J. Keane. *Engineering Design via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, 2008. [43](#), [66](#), [67](#), [92](#), [98](#), [109](#), [113](#)
- [81] R. Frank. The perceptron a perceiving and recognizing automaton. *Cornell Aeronautical Laboratory, Buffalo, NY, USA, Tech. Rep*, pages 85–460, 1957. [11](#)
- [82] R. Franke. A critical comparison of some methods for interpolation of scattered data. Technical report, Monterey, California: Naval Postgraduate School., 1979. [120](#)
- [83] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer Series in Statistics New York, 2001. [30](#)

- 
- [84] A. Garbuno-Inigo, F. A. DiazDelaO, and K. M. Zuev. Transitional annealed adaptive slice sampling for gaussian process hyper-parameter estimation. *International Journal for Uncertainty Quantification*, 6(4):341–359, 2016. [81](#), [115](#), [121](#), [129](#)
- [85] A. Garbuno-Inigo, F. A. DiazDelaO, and K. M. Zuev. Gaussian process hyper-parameter estimation using Parallel Asymptotically Independent Markov Sampling . *Computational Statistics & Data Analysis*, 103:367 – 383, 2016. [43](#), [81](#), [95](#)
- [86] A. Garbuno-Inigo, F. A. DiazDelaO, and K. M. Zuev. Emulation-based history matching. *In preparation*, 2018. [107](#)
- [87] A. Garbuno-Inigo, F. A. DiazDelaO, and K. M. Zuev. Hamiltonian-based Subset Simulation. *In preparation*, 2018. [137](#)
- [88] A. Gelman, G. Roberts, and W. Gilks. Efficient metropolis jumping rules. *Bayesian statistics*, 5:599–608, 1996. [88](#), [147](#)
- [89] A. Gelman, A. Jakulin, M. G. Pittau, and Y. S. Su. A weakly informative default prior distribution for logistic and other regression models. *Annals of Applied Statistics*, 2(4):1360–1383, 2008. [27](#)
- [90] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*, volume 2. CRC press Boca Raton, FL, 2014. [41](#)
- [91] A. Gelman, D. Simpson, and M. Betancourt. The prior can generally only be understood in the context of the likelihood. *arXiv preprint arXiv:1708.07487*, 2017. [27](#)
- [92] C. J. Geyer. Introduction to Markov Chain Monte Carlo. *Handbook of Markov Chain Monte Carlo*, (1990):3–48, 2002. [28](#), [29](#), [142](#)
- [93] R. Ghanem and P. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer, New York Berlin Heidelberg, 1991. Revised Edition 2003, Dover Publications, Mineola, New York. [14](#)
- [94] R. Ghanem, D. Higdon, and H. Owhadi. *Handbook of uncertainty quantification*. Springer, 2017. [14](#)

- [95] M. N. Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, Citeseer, 1998. [44](#), [64](#)
- [96] I. Gilboa. *Theory of Decision Under Uncertainty*. Cambridge University Press, 2009. [3](#)
- [97] D. Ginsbourger, O. Roustant, D. Schuhmacher, N. Durrande, and N. Lenz. On anova decompositions of kernels and gaussian random field paths. In *Monte Carlo and Quasi-Monte Carlo Methods*, pages 315–330. Springer, 2016. [36](#)
- [98] M. Girolami, B. Calderhead, and S. a. Chin. Riemannian Manifold Hamiltonian Monte Carlo. page 35, 2009. [145](#)
- [99] M. Goldstein and N. Huntley. Bayes linear emulation, history matching, and forecasting for complex computer simulators. *Handbook of Uncertainty Quantification*, pages 1–24, 2016. [1](#), [16](#), [17](#)
- [100] M. Goldstein and D. Wooff. *Bayes Linear Statistics, Theory and Methods*. Wiley Series in Probability and Statistics. Wiley, 2007. [16](#), [112](#)
- [101] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996. [64](#)
- [102] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep Learning*. MIT press Cambridge, 2016. [11](#), [12](#)
- [103] R. B. Gramacy and N. G. Polson. Particle learning of Gaussian process models for sequential design and optimization. *Journal Of Computational And Graphical Statistics*, 20(1):18, 2009. [46](#), [92](#), [97](#)
- [104] P. Green and S. Maskell. Estimating the parameters of dynamical systems from big data using sequential monte carlo samplers. *Mechanical Systems and Signal Processing*, 93:379–396, 2017. [29](#), [46](#), [91](#)
- [105] E. P. Gritit, T. Gneiting, V. J. Berrocal, and N. a. Johnson. The continuous ranked probability score for circular variables and its application to mesoscale forecast ensemble verification. *Quarterly Journal of the Royal Meteorological Society*, 132(621C):2925–2942, 2006. [93](#), [94](#)

- 
- [106] M. D. Guenov, M. Nunez, A. Molina-Cristóbal, V. C. Datta, and A. Riaz. Aircadiaan interactive tool for the composition and exploration of aircraft computational studies at early design stage. In *29th Congress of the International Council of the Aeronautical Sciences*, 2014. [130](#)
- [107] B. Haaland and P. Z. Qian. Accurate emulators for large-scale computer experiments. *The Annals of Statistics*, 39(6):2974–3002, 2011. [92](#), [94](#)
- [108] H. Haario, E. Saksman, and J. Tamminen. An Adaptive Metropolis Algorithm. *Bernoulli*, 7(2):223–242, 2001. [60](#)
- [109] J. Y. Halpern. *Reasoning About Uncertainty*. MIT Press, 2017. [3](#)
- [110] R. Hankin. Introducing BACCO, an R bundle for Bayesian analysis of computer code output. *Journal of Statistical Software*, 14(16), 2005. [45](#)
- [111] W. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 82:711–732, 1970. [29](#), [163](#)
- [112] T. Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37(2):185–194, 1995. [87](#)
- [113] D. Higdon, M. Kennedy, J. C. Cavendish, J. A. Cafeo, and R. D. Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466, 2004. [4](#)
- [114] M. Hoffman and A. Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:30, 2014. [146](#)
- [115] M. Hoffman, E. Brochu, and N. de Freitas. Portfolio allocation for bayesian optimization. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 327–336. AUAI Press, 2011. [124](#), [135](#)
- [116] M. Hohenbichler and R. Rackwitz. Non-normal dependent vectors in structural safety. *Journal of the Engineering Mechanics Division*, 107(6):1227–1238, 1981. [138](#)

- [117] P. B. Holden, N. R. Edwards, J. Hensman, and R. D. Wilkinson. Abc for climate: dealing with expensive simulators. *arXiv preprint*, 2015. [113](#)
- [118] K. Hornik, F. Leisch, and A. Zeileis. JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of DSC*, volume 2, pages 1–1, 2003. [28](#)
- [119] P. Hristov, F. DiazDelaO, K. Kubiak, and U. Farooq. Adaptive emulation-based reliability analysis. In *2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*, 2017. [159](#)
- [120] D. E. Hudson. Dynamic tests of full-scale structures. *Journal of Engineering Mechanics Division*, 103(6):1141–1157, 1977. [162](#)
- [121] E. Jaynes and G. Bretthorst. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003. [162](#)
- [122] H. Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, pages 453–461, 1946. [26](#)
- [123] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998. [109](#)
- [124] A. A. Kalaitzis and N. D. Lawrence. A simple approach to ranking differentially expressed gene expression time courses through Gaussian process regression. *BMC bioinformatics*, 12(1):1, 2011. [43](#), [108](#)
- [125] L. Katafygiotis and S. Cheung. Spherical subset simulation ( $s^3$ ) for solving nonlinear dynamical reliability problems. *Intl. Journal of Reliability and Safety (in print)*, 2006. [144](#)
- [126] L. Katafygiotis and K. Zuev. Estimation of small failure probabilities in high dimensions by adaptive linked importance sampling. In *ECCOMAS thematic conference*, pages 1–12, 2007. [86](#)



- 
- [127] L. Katafygiotis and K. Zuev. Geometric insight into the challenges of solving high-dimensional reliability problems. *Probabilistic Engineering Mechanics*, 23(2-3):208–218, April-July 2008. 5th International Conference on Computational Stochastic Mechanics. [28](#), [138](#), [143](#), [147](#), [154](#), [164](#)
- [128] L. S. Katafygiotis and J. L. Beck. Updating models and their uncertainties ii: Model identifiability. *Journal of Engineering Mechanics, ASCE*, 124(4):463–467, 1998. [163](#)
- [129] L. S. Katafygiotis and H.-F. Lam. Tangential-projection algorithm for manifold representation in unidentifiable model updating problems. *Earthquake engineering & structural dynamics*, 31(4):791–812, 2002. [163](#)
- [130] M. C. Kennedy and A. O’Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, Aug. 2001. [4](#), [43](#), [44](#), [45](#), [91](#), [93](#), [108](#), [110](#)
- [131] M. C. Kennedy and A. O’Hagan. Supplementary details on Bayesian Calibration of Computer Models. Technical report, 2001. [44](#)
- [132] S. Kirkpatrick et al. Optimization by simulated annealing. *Journal of statistical physics*, 34(5-6):975–986, 1983. [55](#)
- [133] J. Kohonen and J. Suomela. Lessons learned in the challenge: Making predictions and scoring them. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3944 LNAI:95–116, 2006. [93](#)
- [134] K. Konakli and B. Sudret. Uncertainty quantification in high-dimensional spaces with low-rank tensor approximations. In *1st International Conference on Uncertainty Quantification in Computational Sciences and Engineering*, 2015. [7](#), [14](#)
- [135] K. Konakli and B. Sudret. Low-rank tensor approximations for reliability analysis. In *12th International Conference on Applications of Statistics and Probability in Civil Engineering (ICASP12)*, 2015. [14](#)
- [136] K. Konakli and B. Sudret. Global sensitivity analysis using low-rank tensor approximations. *Reliability Engineering & System Safety*, 156:64–83, 2016. [14](#), [15](#)

- [137] D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Southern African Institute of Mining and Metallurgy*, 52(6):119–139, 1951. [4](#)
- [138] J. Kruschke. *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press, 2014. [41](#)
- [139] R. C. Kuczera and Z. P. Mourelatos. On estimating the reliability of multiple failure region problems using approximate metamodels. *Journal of Mechanical Design*, 131(12), 2009. [116](#), [119](#)
- [140] P.-S. Laplace. *A Philosophical Essay on Probabilities*,. New York: Dover Publications Inc., 1814. [24](#)
- [141] H.-S. Li and S.-K. Au. Design optimization using Subset Simulation algorithm. *Structural Safety*, 32(6):384–392, 2010. [164](#)
- [142] F. Liu, M. Bayarri, J. Berger, et al. Modularization in bayesian analysis, with emphasis on analysis of computer models. *Bayesian Analysis*, 4(1): 119–150, 2009. [35](#)
- [143] J. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. Springer, 2001. [29](#)
- [144] M. H. Liu and G. T. Zheng. Improved component-mode synthesis for nonclassically damped systems. *AIAA Journal*, 46(5):1160–1168, 2008. [45](#)
- [145] Z. Lv, Z. Lu, and P. Wang. A new learning function for kriging and its applications to solve reliability problems in engineering. *Computers & Mathematics with Applications*, 70(5):1182–1197, 2015. [109](#), [118](#)
- [146] D. J. MacKay. Introduction to Monte Carlo methods. In *Learning in graphical models*, pages 175–204. Springer, 1998. [45](#)
- [147] D. J. C. MacKay. Hyperparameters: Optimize, or integrate out? *Maximum entropy and Bayesian methods*, pages 43–59, 1996. [54](#)
- [148] D. Malakoff. Bayes offers a 'new' way to make sense of numbers. *Science*, 286(5444):1460–1464, 1999. [162](#)

- 
- [149] E. Marinari and G. Parisi. Simulated tempering: a new monte carlo scheme. *EPL (Europhysics Letters)*, 19(6):451, 1992. [102](#)
- [150] C. Maschio and D. J. Schiozer. Bayesian history matching using artificial neural network and Markov Chain Monte Carlo. *Journal of Petroleum Science and Engineering*, 123:62–71, Nov. 2014. [7](#)
- [151] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943. [11](#)
- [152] R. McElreath. *Statistical Rethinking: A Bayesian Course with Examples in R and Stan*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2016. [41](#)
- [153] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44:335–341, 1949. [29](#)
- [154] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *J. Chem. Physics*, 21(6):1087–1092, 1953. [29](#), [163](#)
- [155] T. P. Minka. Deriving quadrature rules from gaussian processes. Technical report, 2000. [40](#)
- [156] A. Mira. On Metropolis-Hastings algorithms with delayed rejection. *Metron*, 59(3-4):231–241, 2001. [46](#), [62](#)
- [157] H. Mohasel Afshar and J. Domke. Reflection, refraction, and hamiltonian monte carlo. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3007–3015. Curran Associates, Inc., 2015. [146](#)
- [158] I. Murray and R. P. Adams. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2010. [85](#)
- [159] I. Murray, R. P. Adams, and D. J. MacKay. Elliptical slice sampling. In *AISTATS*, volume 13, pages 541–548, 2010. [85](#)

- [160] R. M. Neal. Probabilistic inference using Markov Chain Monte Carlo methods. *Technical Report, University of Toronto*, pages 1–144, 1993. [45](#)
- [161] R. M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6(4):353–366, 1996. [46](#), [102](#)
- [162] R. M. Neal. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. *Technical Report, University of Toronto*, (9702):1–24, 1997. [58](#), [92](#)
- [163] R. M. Neal. Regression and classification using Gaussian process priors. *Bayesian Statistics*, 6, 1998. [45](#), [55](#), [85](#)
- [164] R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001. [45](#), [46](#)
- [165] R. M. Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003. [55](#), [81](#), [82](#), [84](#), [88](#)
- [166] R. M. Neal. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*, pages 113–162. 2011. [45](#), [145](#), [146](#)
- [167] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012. [12](#)
- [168] T. Nilson and A. Kuusk. A reflectance model for the homogeneous plant canopy and its inversion. *Remote Sensing of Environment*, 27(2):157–167, 1989. [74](#), [92](#)
- [169] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2004. [44](#)
- [170] B. J. Oakley and A. O. Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. pages 769–784, 2002. [4](#)
- [171] J. Oakley. *Bayesian Uncertainty Analysis for Complex Computer Codes*. PhD thesis, 1999. [1](#), [44](#), [49](#), [51](#), [54](#)
- [172] J. Oakley. Eliciting Gaussian process priors for complex computer codes. *Journal of the Royal Statistical Society Series D: The Statistician*, 51(1): 81–97, 2002. [52](#)

- 
- [173] J. E. Oakley and A. O'Hagan. Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, Aug. 2004. [4](#)
- [174] A. O'Hagan. Polynomial Chaos: A Tutorial and Critique from a Statistician's Perspective. *Tonyohagan.Co.Uk*, pages 1–16, 2011. [13](#), [14](#)
- [175] N. Owen, P. Challenor, P. Menon, and S. Bennani. Comparison of surrogate-based uncertainty quantification methods for computationally expensive simulators. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):403–435, 2017. [4](#), [14](#)
- [176] C. Papadimitriou, J. L. Beck, and L. S. Katafygiotis. Updating robust reliability using structural test data. *Probabilistic Engineering Mechanics*, 16(2):103–113, 2001. [162](#)
- [177] V. Papadopoulos, D. G. Giovanis, N. D. Lagaros, and M. Papadrakakis. Accelerated subset simulation with neural networks for reliability analysis. *Computer Methods in Applied Mechanics and Engineering*, 223:70–80, 2012. [144](#)
- [178] I. Papaioannou, W. Betz, K. Zwirgmaier, and D. Straub. Mcmc algorithms for subset simulation, September 2014. [138](#), [144](#), [153](#)
- [179] R. Paulo. Default priors for Gaussian processes. *Annals of Statistics*, 33(2): 556–582, 2005. [52](#), [98](#)
- [180] M. Plummer, N. Best, K. Cowles, and K. Vines. Coda: Convergence diagnosis and output analysis for mcmc. *R News*, 6(1):7–11, 2006. [29](#)
- [181] H. Poincaré. *Calcul des probabilités: leçons professées pendant le deuxième semestre 1893-1894*. Gauthier-Villars, 1896. [31](#)
- [182] F. Pukelsheim. The three sigma rule. *The American Statistician*, 48(2): 88–91, 1994. [17](#), [112](#)
- [183] W. Qi, Z. Lu, and C. Zhou. New Topology Optimization Method for Wing Leading-Edge Ribs. *Journal of Aircraft*, 48(5):1741–1748, sep 2011. [164](#)

- [184] P. Ranjan, D. Bingham, and G. Michailidis. Sequential experiment design for contour estimation from complex computer codes. *Technometrics*, 50(4): 527–541, 2008. [109](#), [117](#), [153](#)
- [185] P. Ranjan, R. Haynes, and R. Karsten. A computationally stable approach to Gaussian process interpolation of deterministic computer simulation data. *Technometrics*, 53(4):366–378, 2011. [46](#), [64](#), [66](#), [82](#), [92](#)
- [186] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. [35](#), [36](#), [41](#), [43](#), [44](#), [108](#)
- [187] C. T. Richard. The algebra of probable inference, 1961. [162](#)
- [188] C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Science & Business Media, 2004. [28](#), [29](#), [45](#), [163](#)
- [189] S. M. Ross. *Simulation*. 2013. [29](#)
- [190] J. Sacks, W. J. Welch, J. S. B. Mitchell, and P. W. Henry. Design and Experiments of Computer Experiments, 1989. [4](#), [131](#)
- [191] J. M. Salter and D. Williamson. A comparison of statistical emulation methodologies for multi-wave calibration of environmental models. *Environmetrics*, 27(8):507–523, 2016. env.2405. [107](#), [116](#), [123](#), [124](#)
- [192] J. Salvatier, T. V. Wiecki, and C. Fonnesbeck. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016. [28](#), [139](#), [151](#)
- [193] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer Science & Business Media, 2013. [4](#)
- [194] L. Savage. *The Foundations of Statistics*. Dover Books on Mathematics Series. Dover Publications, 1972. [24](#), [41](#)
- [195] J. Schneider and S. Kirkpatrick. *Stochastic Optimization*. Scientific Computation. Springer Berlin Heidelberg, 2007. [46](#)
- [196] G. Schuëller, H. Pradlwarter, and P. Koutsourelakis. A critical appraisal of reliability estimation procedures for high dimensions. *Probabilistic Engineering Mechanics*, 19(4):463–474, 2004. [164](#)

- 
- [197] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017. [11](#)
- [198] D. Simpson, H. Rue, A. Riebler, T. G. Martins, S. H. Sørbye, et al. Penalising model component complexity: A principled, practical approach to constructing priors. *Statistical science*, 32(1):1–28, 2017. [27](#)
- [199] R. C. Smith. *Uncertainty Quantification: Theory, Implementation, and Applications*, volume 12. SIAM, 2013. [2](#), [7](#), [108](#)
- [200] A. J. Smola and B. Schölkopf. *Learning with kernels*, volume 4. MIT Press, 1998. [41](#)
- [201] I. M. Sobol’. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel’noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967. [103](#)
- [202] C. Soize. Construction of probability distributions in high dimension using the maximum entropy principle: Applications to stochastic processes, random fields and random matrices. *International Journal for Numerical Methods in Engineering*, 76(10):1583–1611, 2008. [145](#)
- [203] J. Song and W.-H. Kang. System reliability and sensitivity under statistical dependence by matrix-based system reliability method. *Structural Safety*, 31(2):148 – 156, 2009. Risk Acceptance and Risk Communication, Risk Acceptance and Risk Communication. [164](#)
- [204] M. L. Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012. [4](#), [36](#), [40](#), [41](#)
- [205] D. Straub. Reliability updating with equality information. *Probabilistic Engineering Mechanics*, 26(2):254–258, 2011. [165](#)
- [206] D. Straub and I. Papaioannou. Bayesian Updating with Structural Reliability Methods. (May):1–34, 2014. [5](#), [159](#), [161](#), [165](#), [166](#), [167](#), [168](#)
- [207] D. Straub, I. Papaioannou, and W. Betz. Bayesian analysis of rare events. *Journal of Computational Physics*, 314:538–556, 2016. [166](#)

- [208] A. Stuart and A. Teckentrup. Posterior consistency for gaussian process approximations of bayesian posterior distributions. *Mathematics of Computation*, 87(310):721–753, 2018. [129](#)
- [209] A. A. Taflanidis and J. L. Beck. Stochastic Subset Optimization for optimal reliability problems. *Probabilistic Engineering Mechanics*, 23(2):324–338, 2008. [46](#), [81](#)
- [210] A. A. Taflanidis and J. L. Beck. An efficient framework for optimal robust stochastic system design using stochastic simulation. *Computer Methods in Applied Mechanics and Engineering*, 198(1):88–101, 2008. [46](#), [81](#)
- [211] A. A. Taflanidis and J. L. Beck. Life-cycle cost optimal design of passive dissipative devices. *Structural Safety*, 31(6):508 – 522, 2009. Optimization under Uncertainty with Emphasis on Structural Applications Special Issue: Optimization. [164](#)
- [212] Z. Tavassoli, J. N. Carter, and P. R. King. An analysis of history matching errors. *Computational Geosciences*, 9(2):99–123, 2005. [123](#)
- [213] D. Tran, A. Kucukelbir, A. B. Dieng, M. Rudolph, D. Liang, and D. M. Blei. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*, 2016. [28](#), [151](#)
- [214] M. Vanik, J. Beck, and S.-K. Au. Bayesian probabilistic approach to structural health monitoring. *Journal of Engineering Mechanics*, 126:738–745, 2000. [183](#)
- [215] E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and inference*, 140(11):3088–3095, 2010. [109](#)
- [216] I. Vernon, M. Goldstein, and R. G. Bower. Galaxy formation: a Bayesian uncertainty analysis. *Bayesian Analysis*, 5(4):619–669, Dec. 2010. [16](#), [17](#), [54](#), [110](#), [111](#), [123](#)
- [217] I. Vernon, M. Goldstein, R. Bower, et al. Galaxy formation: Bayesian history matching for the observable universe. *Statistical Science*, 29(1):81–90, 2014. [107](#), [112](#), [126](#)



- 
- [218] J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509, 2009. [116](#)
- [219] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman & Hall, London, 1991. [3](#)
- [220] Z. Wang, M. Broccardo, and J. Song. Hamiltonian monte carlo methods for subset simulation in reliability analysis. *arXiv preprint arXiv:1706.01435*, 2017. [146](#), [153](#)
- [221] R. D. Wilkinson. Accelerating ABC methods using Gaussian processes. *AISTATS*, pages 1015–1023, 2014. [43](#)
- [222] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. *Advances in Neural Information Processing Systems*, pages 514–520, 1996. [44](#), [45](#), [55](#), [85](#)
- [223] D. Williamson and I. Vernon. Efficient uniform designs for multi-wave computer experiments. *arXiv preprint arXiv:1309.3520*, 2013. [113](#), [115](#), [116](#)
- [224] D. Williamson, M. Goldstein, L. Allison, A. Blaker, P. Challenor, L. Jackson, and K. Yamazaki. History matching for exploring and reducing climate model parameter space using observations and a large perturbed physics ensemble. *Climate dynamics*, 41(7-8):1703–1729, 2013. [113](#)
- [225] T. Yeh, T. Uvieghara, J. Jennings, C. Chen, F. Alpak, F. Tendo, et al. A practical workflow for probabilistic history matching and forecast uncertainty quantification: Application to a deepwater west africa reservoir. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2016. [114](#)
- [226] K. M. Zuev and J. L. Beck. Global optimization using the Asymptotically Independent Markov Sampling method. *Computers & Structures*, 126: 107–119, 2013. [46](#), [55](#), [58](#), [59](#), [60](#), [61](#), [78](#), [81](#), [90](#), [114](#), [121](#)
- [227] K. M. Zuev and L. S. Katafygiotis. Modified Metropolis-Hastings algorithm with delayed rejection. *Probabilistic Engineering Mechanics*, 26(3):405–412, 2011. [46](#), [62](#), [138](#), [141](#), [144](#)

- [228] K. M. Zuev, J. L. Beck, S. K. Au, and L. S. Katafygiotis. Bayesian post-processor and other enhancements of Subset Simulation for estimating failure probabilities in high dimensions. *Computers and Structures*, 92-93:283–296, 2012. [152](#)